

Optimal Key Consensus in Presence of Noise*

Zhengzhong Jin[†] Yunlei Zhao[‡]

Abstract

In this work, we introduce and formalize a new primitive, referred to as key consensus (KC), and its asymmetric variant AKC, for communicating parties reaching consensus from close values. Then we find efficient upper bounds for any KC and AKC schemes. The bounds are particularly instrumental in choosing parameters towards different optimization goals. KC and AKC are fundamental to lattice based cryptography, in the sense that a list of cryptographic primitives based on LWE or Ring-LWE (including key exchange, public key encryption, oblivious transfer, and more) can be modularly constructed from them. As a conceptual contribution, this much simplifies the design and analysis of these cryptosystems in the future.

We then design and analyze highly practical KC and AKC schemes within a general framework. The correctness constraint on parameters of our schemes is almost the same as the efficiency upper bound. The structure generalization and tightness allow us to choose parameters towards optimal balance among security, computational cost, bandwidth, consensus range, and error rate. When applied to LWE or RLWE based cryptosystems, generally speaking, by carefully choosing parameters they can lead to more practical schemes of key exchange and CPA-secure public key encryption.

*A preliminary version of this work is in submission to IEEE S&P 2017. Compared with the S&P submission version, the following main revisions (besides some readability and presentation refinements) are made in this version: Firstly, the technique of cutting off some least significant bits is introduced and analyzed for LWE-based protocols; Secondly, a brief note on Lizard [CKLS16] is made; Finally, the codes of OKCN-LWE are integrated into liboqs [SM16] on Github. This research was supported in part by NSFC Grant No. U1536205.

[†]School of Mathematical Sciences, Fudan University, Shanghai, China. zzjin13@fudan.edu.cn. Zhengzhong Jin is now an undergraduate.

[‡]School of Computer Science, Fudan University, Shanghai, China. ylzhaoy@fudan.edu.cn

Contents

1	Introduction	3
1.1	Our Contributions	4
1.2	Future Works	5
2	Preliminaries	6
2.1	The LWE and RLWE problems	7
3	Key Consensus with Noise	8
3.1	Efficiency Upper Bound of KC	9
3.2	Construction and Analysis of OKCN	11
3.2.1	Correctness and Security of OKCN	12
3.2.2	Special Parameters, and Performance Speeding-Up	13
4	Asymmetric Key Consensus with Noise	15
4.1	Efficiency Upper Bound of AKC	16
4.2	Construction and Analysis of AKCN	18
4.2.1	Correctness and Security of AKCN	18
4.2.2	Special Parameters and Performance Speeding-Up	19
5	LWE-Based Key Exchange from KC and AKC	20
5.1	Security Analysis	22
5.2	Noise Distributions and Correctness	27
5.2.1	Binary Distribution	29
5.2.2	Discrete distribution	30
5.3	Instantiations, and Comparisons with Frodo	31
5.4	Integration into liboqs, and Benchmark	35
6	RLWE-Based Key Exchange from KC and AKC	37
6.1	AKCN-RLWE with Negligible Error Rate	39
6.1.1	Overview of NewHope	39
6.1.2	Construction and Analysis of AKCN-4:1	41
6.2	Instantiations, and Comparison with NewHope	42
7	Applications to PKE, OT, Key Transport, and TLS 1.3	43
A	On the Codes of Evaluating Error Rates of KE from OKCN and AKCN	49
B	Consensus Mechanism of Frodo	49
C	Consensus Mechanism of NewHope	50
D	A Note on Lizard	51

1 Introduction

Most public-key cryptosystems currently in use, based on the hardness of solving (elliptic curve) discrete logarithm or factoring large integers, will be broken, if large-scale quantum computers are ever built. The arrival of such quantum computers is now believed by many scientists to be merely a significant engineering challenge, and is estimated by engineers at IBM to be within the next two decades or so. Historically, it has taken almost two decades to deploy the modern public key cryptography infrastructure. Therefore, regardless of whether we can estimate the exact time of the arrival of the quantum computing era, we must begin now to prepare our information security systems to be able to resist quantum computing [NIS]. In addition, for the content we want to protect over a period of 15 years or longer, it becomes necessary to switch to post-quantum cryptography today. This has been recognized not only by the cryptography research community, but also by standardization bodies and leading information companies, for example, NSA [NSA], NIST [NIS], U.K. CESG [CESG], the Tor project [Nic], and Google [Mat].

As noted in [ADPS16, AJS16], in the majority of contexts the most critical asymmetric primitive to upgrade to post-quantum security is ephemeral key exchange (KE). KE plays a central role in modern cryptography, which bridges public-key cryptography and symmetric-key cryptography and can, in turn, be used to build CPA-secure public-key encryption (PKE) as well as CCA-secure PKE in the random oracle (RO) model via the FO-transformation [FO13], and oblivious transfer [GKM⁺00], and more. U.K. CESG has also expressed their preference for post-quantum algorithms (in particular, post-quantum KE schemes) over quantum technologies “such as Quantum Key Distribution” to counter the threat of quantum computing [CESG].

Lattice-based cryptography is among the major mathematical approaches to achieving security resistant to quantum attacks. For cryptographic usage, compared with the classic hard lattice problems such as SVP and CVP, the learning with errors (LWE) problem is proven to be much more versatile [Reg09]. Nevertheless, LWE-based cryptosystems are usually less efficient, which was then resolved by the introduction of the ring-LWE (RLWE) problem [LPR13a]. In recent years, large numbers of impressive works are developed from LWE and RLWE, with (ephemeral) key exchange and public-key encryption being the study focus of this work [JD12, Pei14, BCNS15, ADPS16, BCD⁺16, Reg09, GPV08, LP10, LPR13a, LPR13b, PG13]. For an excellent survey of lattice-based cryptography, the reader is referred to [Pei16].

Some celebrating progresses on achieving practical LWE- and RLWE-based key exchange are made in recent years. The performance of RLWE-based key exchange is significantly improved with NewHope [ADPS16], which stands for the most practical RLWE-based key exchange protocol up to now. But NewHope does not directly yield a CPA-secure PKE scheme. Compared to LWE, the additional ring structure of RLWE helps to improve the efficiency of cryptosystems, but the concrete hardness of RLWE remains less clear. The work [BCD⁺16] proposes a key exchange protocol Frodo only based on LWE, and demonstrates that LWE-based key exchange can be practical as well. Nevertheless, bandwidth of Frodo is relatively large, as Frodo uses about 22kB bandwidth for its recommended parameter set. In addition, Frodo has relatively larger error rates, and also cannot be directly used for PKE. One of the main contributions in these works [ADPS16, BCD⁺16, PG13], among others, is the improvement and generalization of Peikert’s reconciliation mechanism [Pei14], which helps the two communicating parties reaching consensus from close values obtained by exchanging their LWE/RLWE samples. Whether further improvements on LWE- and RLWE-based key exchange, as well as CPA-secure PKE, can be achieved remains an interesting question of practical significance.

1.1 Our Contributions

In this work, we introduce and formalize a new primitive, referred to as key consensus (KC), and its asymmetric variant AKC, for two communicating parties reaching consensus from close values obtained by some secure information exchange, such as exchanging their LWE/RLWE samples. We then discover upper bounds of parameters for any KC or AKC. KC and AKC are fundamental to lattice based cryptography, in the sense that a list of cryptographic primitives based on LWE or RLWE (including key exchange, public-key encryption, oblivious transfer, and more) can be modularly constructed from them.

We then design and analyze both general and highly practical KC and AKC schemes. Our KC and AKC schemes are optimal in a sense of achieving optimal balance among security, (computational and bandwidth) efficiency, error rate, and operation simplicity. Firstly, the correctness constraints on parameters are almost the same as the upper bounds we discovered. Secondly, the generality of our schemes allows us to take optimal balance among parameters, and chooses parameters towards different goals. Thirdly, the operations involved are simple.

When applied to LWE-based cryptosystems, they can result in more practical schemes of key exchange, CPA-secure public-key encryption. For LWE-based key exchange, to further save bandwidth, we cut off some least significant bits of LWE samples, and provide a delicate analysis of error rate. These improvements result in a more efficient key exchange scheme with 18.58kB bandwidth, which provides at least 128 bit post-quantum security.¹ When applied to RLWE-based cryptosystems, to the best of our knowledge, as we shall see in Section 6.2, they can lead to more practical schemes of CPA-secure public-key encryption, and AKC-based key exchange (KC-based key exchange is somewhat incomparable with NewHope). We also discuss the applications of KC and AKC to authenticated key exchange, to TLS1.3, and to CCA-secure PKE based on LWE and RLWE.

1.2 Future Works

In this work, we have mainly focused upon the instantiations of KC/AKC based cryptosystems from some relatively standard LWE and RLWE assumptions. That is, the close values are obtained by generating and exchanging LWE or RLWE samples. Moreover, the samples are generated and exchanged in a symmetric way with the same LWE or RLWE problem used by both of the communicating peers. However, the generality of our framework allows for more instantiations with other variants of LWE and RLWE, and allows for more possible balance approaches, which are left for future works.

For example, we note that the efficiency and error rates of these cryptosystems can be simply improved with binary-LWE, spLWE [CHKLS16], LWR [BGMRR16] or spLWR [CKLS16] (over small modulus), etc, but the concrete hardness of some of these variants are less clear at the current stage. When applied to PKE, the samples can be generated in an asymmetric way in order to achieve better trade-offs among security, efficiency and error rate according to the application scenarios. Specifically, the samples in public key are generated with harder LWE/RLWE problems (in order to ensure security with any polynomial number of samples exposed to adversary), but the samples in the ephemeral ciphertext can be generated binary-LWE, spLWE, LWR or spLWR (over small modulus), etc. We note that a PKE scheme instantiated from our AKCN scheme presented in Algorithm 4 has

¹The at least 128 bit post-quantum security is evaluated without taking cutting off some least significant bits into account. We stress that, cutting off some least bits of LWE samples can further improve the actual security guarantee in reality, which, however, we do not know how to quantitatively measure the extra security gains now.

already been analyzed recently in [CKLS16], where the two close values are derived from generating and exchanging spLWE and spLWR samples in an asymmetric way.

Another important research direction is to further improve the coding methodologies of KC and AKC, by seeking for more advanced sphere packing techniques from lattice and glue theory. From the point of our view, the difficulty mainly lies in the balance between bandwidth gain and operation simplicity.

2 Preliminaries

A string or value α means a binary one, and $|\alpha|$ is its binary length. For any real number x , $\lfloor x \rfloor$ denotes the largest integer that less than or equal to x , and $\lceil x \rceil = \lfloor x + 1/2 \rfloor$. For any positive integers a and b , denote by $\text{lcm}(a, b)$ the least common multiple of them. For any $i, j \in \mathbb{Z}$ such that $i < j$, denote by $[i, j]$ the set of integers $\{i, i+1, \dots, j-1, j\}$. For any positive integer t , we let \mathbb{Z}_t denote $\mathbb{Z}/t\mathbb{Z}$. The elements of \mathbb{Z}_t are represented, by default, as $[0, t-1]$. Nevertheless, sometimes, \mathbb{Z}_t is explicitly specified to be represented as $[-\lfloor (t-1)/2 \rfloor, \lfloor t/2 \rfloor]$.

If S is a finite set then $|S|$ is its cardinality, and $x \leftarrow S$ is the operation of picking an element uniformly at random from S . For two sets $A, B \subseteq \mathbb{Z}_q$, define $A + B \triangleq \{a + b \mid a \in A, b \in B\}$. For an additive group $(G, +)$, an element $x \in G$ and a subset $S \subseteq G$, denote by $x + S$ the set containing $x + s$ for all $s \in S$. For a set S , denote by $\mathcal{U}(S)$ the uniform distribution over S . For any random variable X over \mathbb{R} , denote $\text{Supp}(X) = \{x \in \mathbb{R} \mid \Pr[X = x] > 0\}$.

We use standard notations and conventions below for writing probabilistic algorithms, experiments and interactive protocols. If \mathcal{D} denotes a probability distribution, $x \leftarrow \mathcal{D}$ is the operation of picking an element according to \mathcal{D} . If α is neither an algorithm nor a set then $x \leftarrow \alpha$ is a simple assignment statement. If A is a probabilistic algorithm, then $A(x_1, x_2, \dots; r)$ is the result of running A on inputs x_1, x_2, \dots and coins r . We let $y \leftarrow A(x_1, x_2, \dots)$ denote the experiment of picking r at random and letting y be $A(x_1, x_2, \dots; r)$. By $\Pr[R_1; \dots; R_n : E]$ we denote the probability of event E , after the ordered execution of random processes R_1, \dots, R_n .

We say that a function $f(\lambda)$ is *negligible*, if for every $c > 0$ there exists an λ_c such that $f(\lambda) < 1/\lambda^c$ for all $\lambda > \lambda_c$. Two distribution ensembles $\{X(\lambda, z)\}_{\lambda \in N, z \in \{0,1\}^*}$ and $\{Y(\lambda, z)\}_{\lambda \in N, z \in \{0,1\}^*}$ are computationally indis-

tinguishable, if for any probabilistic polynomial-time (PPT) algorithm D , and for sufficiently large λ and any $z \in \{0, 1\}^*$, it holds $|\Pr[D(\lambda, z, X) = 1] - \Pr[D(\lambda, z, Y) = 1]|$ is negligible in λ .

2.1 The LWE and RLWE problems

Given positive *continuous* $\alpha > 0$, define the real Gaussian function $\rho_\alpha(x) \triangleq \exp(-x^2/2\alpha^2)/\sqrt{2\pi\alpha^2}$ for $x \in \mathbb{R}$. Let $D_{\mathbb{Z},\alpha}$ denote the 1-dimensional *discrete* Gaussian distribution over \mathbb{Z} , which is determined by its probability density function $D_{\mathbb{Z},\alpha}(x) \triangleq \rho_\alpha(x)/\rho_\alpha(\mathbb{Z})$, $x \in \mathbb{Z}$. Finally, let $D_{\mathbb{Z}^n,\alpha}$ denote the n -dimensional *spherical* discrete Gaussian distribution over \mathbb{Z}^n , where each coordinate is drawn *independently* from $D_{\mathbb{Z},\alpha}$.

Given positive integers n and q that are both polynomial in the security parameter λ , an integer vector $\mathbf{s} \in \mathbb{Z}_q^n$, and a probability distribution χ on \mathbb{Z}_q , let $A_{q,\mathbf{s},\chi}$ be the distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ obtained by choosing $\mathbf{a} \in \mathbb{Z}_q^n$ uniformly at random, and an error term $e \leftarrow \chi$, and outputting the pair $(\mathbf{a}, b = \mathbf{a}^T \mathbf{s} + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. The error distribution χ is typically taken to be the discrete Gaussian probability distribution $D_{\mathbb{Z},\alpha}$ defined previously; However, as suggested in [BCD⁺16] and as we shall see in Section 5.2, other alternative distributions of χ can be taken. Briefly speaking, the (decisional) *learning with errors* (LWE) assumption [Reg09] says that, for sufficiently large security parameter λ , no probabilistic polynomial-time (PPT) algorithm can distinguish, with non-negligible probability, $A_{q,\mathbf{s},\chi}$ from the uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$. This holds even if \mathcal{A} sees polynomially many samples, and even if the secret vector \mathbf{s} is drawn randomly from χ^n [ACPS09].

For the positive integer m that is polynomial in the security parameter λ , let $n \triangleq \varphi(m)$ denote the totient of m , and $\mathcal{K} \triangleq \mathbb{Q}(\zeta_m)$ be the number field obtained by adjoining an abstract element ζ_m satisfying $\Phi_m(\zeta_m) = 0$, where $\Phi_m(x) \in \mathbb{Z}[x]$ is the m -th cyclotomic polynomial of degree n . Moreover, let $\mathcal{R} \triangleq \mathcal{O}_{\mathcal{K}}$ be the ring of integers in \mathcal{K} . Finally, given a positive prime $q = \text{poly}(\lambda)$ such that $q \equiv 1 \pmod{m}$, define the quotient ring $\mathcal{R}_q \triangleq \mathcal{R}/q\mathcal{R}$.

We briefly review the RLWE problem, and its hardness result [LPR13a, LPR13b, DD12]. As we shall see, it suffices in this work to consider a *special* case of the original ring-LWE problem defined in [LPR13a]. Let $n \geq 16$ be a power-of-two and $q = \text{poly}(\lambda)$ be a positive prime such that $q \equiv 1 \pmod{2n}$. Given $\mathbf{s} \leftarrow \mathcal{R}_q$, a sample drawn from the RLWE distribution $A_{n,q,\alpha,\mathbf{s}}$ over $\mathcal{R}_q \times \mathcal{R}_q$ is generated by first choosing $\mathbf{a} \leftarrow \mathcal{R}_q$, $\mathbf{e} \leftarrow D_{\mathbb{Z}^n,\alpha}$, and then outputting $(\mathbf{a}, \mathbf{a} \cdot \mathbf{s} + \mathbf{e}) \in \mathcal{R}_q \times \mathcal{R}_q$. Roughly speaking, the (decisional) RLWE assumption says that, for sufficiently large security parameter λ , no

PPT algorithm \mathcal{A} can distinguish, with non-negligible probability, $A_{n,q,\alpha,\mathbf{s}}$ from the uniform distribution over $\mathcal{R}_q \times \mathcal{R}_q$. This holds even if \mathcal{A} sees polynomially many samples, and even if the secret \mathbf{s} is drawn randomly from the same distribution of the error polynomial \mathbf{e} [DD12, ACPS09]. Moreover, as suggested in [ADPS16] and as we shall see in Section 6.2, alternative distributions for the error polynomials can be taken for the sake of efficiency while without essentially reducing security.

3 Key Consensus with Noise

Before presenting the definition of key consensus (KC) scheme, we first introduce a new function $|\cdot|_t$ relative to arbitrary positive integer $t \geq 1$:

$$|x|_t = \min\{x \bmod t, t - x \bmod t\}, \quad \forall x \in \mathbb{Z}.$$

In the following description, we use $|\sigma_1 - \sigma_2|_q$ to measure the distance between two elements $\sigma_1, \sigma_2 \in \mathbb{Z}_q$. In this work, such a distance is caused by small noises, and is relatively small compared to q .

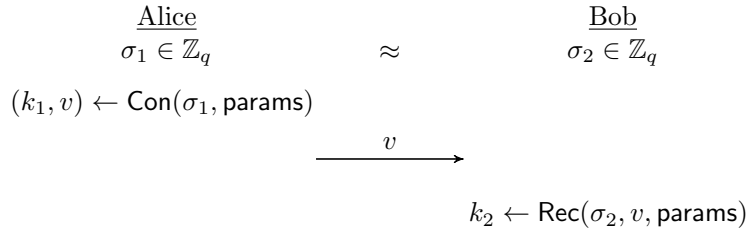


Figure 1: Brief depiction of KC, where $k_1, k_2 \in \mathbb{Z}_m$, $v \in \mathbb{Z}_g$ and $|\sigma_1 - \sigma_2|_q \leq d$.

Definition 3.1. A key consensus scheme $KC = (\text{params}, \text{Con}, \text{Rec})$, briefly depicted in Figure 1, is specified as follows.

- $\text{params} = (q, m, g, d, \text{aux})$ denotes the system parameters, where q, m, g, d are positive integers satisfying $2 \leq m, g \leq q, 0 \leq d \leq \lfloor \frac{q}{2} \rfloor$ (which dominate security, correctness and bandwidth of the KC scheme), and aux denotes some auxiliary values that are usually determined by (q, m, g, d) and could be set to be a special symbol \emptyset indicating “empty”.
- $(k_1, v) \leftarrow \text{Con}(\sigma_1, \text{params})$: On input of $(\sigma_1 \in \mathbb{Z}_q, \text{params})$, the probabilistic polynomial-time conciliation algorithm Con outputs (k_1, v) , where $k_1 \in \mathbb{Z}_m$ is the shared key, and $v \in \mathbb{Z}_g$ is a hint signal that will

be publicly delivered to the communicating peer to help the two parties reach consensus.

- $k_2 \leftarrow \text{Rec}(\sigma_2, v, \text{params})$: On input of $(\sigma_2 \in \mathbb{Z}_q, v, \text{params})$, the deterministic polynomial-time reconciliation algorithm Rec outputs $k_2 \in \mathbb{Z}_m$.

Correctness: A KC scheme is correct, if it holds $k_1 = k_2$ for any $\sigma_1, \sigma_2 \in \mathbb{Z}_q$ such that $|\sigma_1 - \sigma_2|_q \leq d$.

Security: A KC scheme is secure, if k_1 and v are independent, and k_1 is uniformly distributed over \mathbb{Z}_m , whenever $\sigma_1 \leftarrow \mathbb{Z}_q$ (i.e., σ_1 is taken uniformly at random from \mathbb{Z}_q). The probability is taken over the sampling of σ_1 and the random coins used by Con .

3.1 Efficiency Upper Bound of KC

For fixed q, g, d , we expect the two communicating parties to reach as more consensus bits as possible, so the range of consensus key m can be regarded as an indicator of efficiency. The following theorem reveals an upper bound on the range of consensus key of a KC with parameters q, g (parameterize bandwidth), and d (parameterize correctness). Its proof also divulges some intrinsic properties of any *correct* and *secure* KC scheme.

Theorem 3.1. Let $KC = (\text{params}, \text{Con}, \text{Rec})$ be a key consensus scheme, σ_1 and σ_2 are random variables over \mathbb{Z}_q . Suppose that the KC scheme satisfies the following conditions, where the probability is taken over σ_1, σ_2 and the random coins used by Con .

Condition-1. The KC scheme is both correct and secure.

Condition-2. The consensus range m is tight, in the sense that $\text{Supp}(k_1) = \mathbb{Z}_m$.

Condition-3. The distance d is tight: for any $\tilde{\sigma}_1, \tilde{\sigma}_2 \in \mathbb{Z}_q$, if $|\tilde{\sigma}_1 - \tilde{\sigma}_2|_q \leq d$ then $\Pr[\sigma_1 = \tilde{\sigma}_1, \sigma_2 = \tilde{\sigma}_2] > 0$.

Then

$$2md \leq q \left(1 - \frac{1}{g}\right).$$

Before proceeding to prove Theorem 3.1, we first prove the following propositions.

Proposition 3.1. *Given $\text{params} = (q, m, g, d, \text{aux})$ for a correct and secure KC scheme, where d is tight (define in Theorem 3.1, Condition-3), we say a pair $(\sigma_1 \in \mathbb{Z}_q, v \in \mathbb{Z}_g)$ is possible, if $\text{Con}(\sigma_1, \text{params})$ outputs v with positive probability. Then, the value k_1 is fixed w.r.t. the possible (v, σ_1) . That is, for any random coins (r, r') , if $\text{Con}(\sigma_1, \text{params}, r) = (k_1, v)$ and $\text{Con}(\sigma_1, \text{params}, r') = (k'_1, v)$, then $k_1 = k'_1$.*

Proof. For any possible (σ_1, v) , let $\sigma_2 = \sigma_1$, then $|\sigma_1 - \sigma_2|_q = 0 \leq d$. From tightness of d , the event $\sigma_2 = \sigma_1$ happens with positive probability. Then, according to the correctness of KC, we have that $k_1 = k_2 = \text{Rec}(\sigma_2, v) = \text{Rec}(\sigma_1, v)$ where $\sigma_1 = \sigma_2$. However, as Rec is a deterministic algorithm, k_2 is fixed w.r.t. (σ_1, v) . As a consequence, k_1 is also fixed w.r.t. (σ_1, v) , no matter what randomness is used by Con . \square

Proposition 3.2. *Given $\text{params} = (q, m, g, d, \text{aux})$ for a KC scheme, for any $v \in \mathbb{Z}_g$, let S_v be the set containing all σ_1 such that $\text{Con}(\sigma_1, \text{params})$ outputs v with positive probability. Specifically,*

$$S_v = \{\sigma_1 \in \mathbb{Z}_q \mid \Pr[(k_1, v') \leftarrow \text{Con}(\sigma_1, \text{params}) : v' = v] > 0\}.$$

Then, there exists $v_0 \in \mathbb{Z}_g$ such that $|S_{v_0}| \geq q/g$.

Proof. For each $\sigma_1 \in \mathbb{Z}_q$, we run $\text{Con}(\sigma_1, \text{params})$ and get a pair $(k_1, v) \in \mathbb{Z}_m \times \mathbb{Z}_g$ satisfying $\sigma_1 \in S_v$. Then, the proposition is clear by the pigeonhole principle. \square

Proof of Theorem 3.1. From Proposition 3.2, there exists a $v_0 \in \mathbb{Z}_g$ such that $|S_{v_0}| \geq q/g$. Note that, for any $\sigma_1 \in S_{v_0}$, we have that (σ_1, v_0) are possible.

For each $i \in \mathbb{Z}_m$, let K_i denotes the set containing all σ_1 such that $\text{Con}(\sigma_1, \text{params})$ outputs $(k_1 = i, v = v_0)$ with positive probability. From Proposition 3.1, K_i 's form a disjoint partition of S_{v_0} . From the independence between k_1 and v (as we assume the underlying KC is secure) and under the tightness of m (Condition-2), we know $\Pr[k_1 = i \mid v = v_0] = \Pr[k_1 = i] > 0$, and so K_i is non-empty for each $i \in \mathbb{Z}_m$. Now, for each $i \in \mathbb{Z}_m$, denote by K'_i the set containing all $\sigma_2 \in \mathbb{Z}_q$ such that $\text{Rec}(\sigma_2, v_0, \text{params}) = i$. As Rec is deterministic, K'_i 's are well-defined and are disjoint.

From the tightness of d (Condition-3) and the correctness of KC, for every $\sigma_1 \in K_i$, $|\sigma_2 - \sigma_1|_q \leq d$, we have $\sigma_2 \in K'_i$. That is, $K_i + [-d, d] \subseteq K'_i$. Since $K_i + [-d, d]$ contains at least $|K_i| + 2d$ elements, we have $|K_i| + 2d \leq |K'_i|$. When we add up on the both side for all $i \in \mathbb{Z}_m$, then we derive $|S_{v_0}| + 2md \leq q$. By noticing that $|S_{v_0}| \geq q/g$, the theorem is established. \square

Remark: Some comments are in order. Theorem 3.1 divulges an efficiency upper bound on the system parameters of KC schemes, and allows us to take balance on these parameters according to different priorities among security, computational efficiency and bandwidth consumption. When balancing these parameters, we are mainly concerned with the parameters (q, d, m) , with a focus on the parameter q that dominates the security and efficiency of the underlying KC scheme. The parameter g is mainly related to bandwidth. But the bandwidth reduction with a smaller g can be overtaken by the overall efficiency gains with a smaller q . As we shall see in Section 5, when KC is used as a building tool in LWE based cryptosystems, such balance can not only lead to decrement of the bandwidth, improvement of efficiency, but also result in strengthening of security simultaneously.

3.2 Construction and Analysis of OKCN

The key consensus scheme, named “optimally-balanced key consensus with noise (OKCN)”, is presented in Algorithm 1, followed with some explanations for implementation details.

Define $\sigma'_A = \alpha\sigma_1 + e$. Note that it always holds $\sigma'_A < q'$. However, in some rare cases, σ'_A could be a negative value; for example, for the case that $\sigma_1 = 0$ and $e \in [-(\alpha - 1)/2, -1]$. Setting $\sigma_A = \sigma'_A \bmod q'$, in line 4, is to ensure that σ_A is always a non-negative value in $\mathbb{Z}_{q'}$, which can be simply implemented as follows: if $\sigma'_A < 0$ then set $\sigma_A = \sigma'_A + q'$, otherwise set $\sigma_A = \sigma'_A$. Considering potential timing attacks, conditional statement judging whether σ'_A is negative or not can be avoided by a bitwise operation extracting the sign bit of σ'_A . In specific, suppose σ'_A is a 16-bit signed or unsigned integer, then one can code $\sigma_A = \sigma'_A + ((\sigma'_A >> 15) \& 1) * q'$ in C language. The same techniques can also be applied to the calculation in line 11.

In lines 5 and 6, (k_1, v') can actually be calculated simultaneously by a single command *div* in assembly language. In line 11, the floating point arithmetic can be replaced by integer arithmetic. If m is small enough, such as 2 or 3, the slow complex integer division operation can be replaced by relative faster conditional statements.

Algorithm 1 OKCN: Optimally-balanced KC with Noise

```
1: params =  $(q, m, g, d, aux)$ ,  $aux = \{q' = \text{lcm}(q, m), \alpha = q'/q, \beta = q'/m\}$ 
2: procedure CON( $(\sigma_1, \text{params})$ )  $\triangleright \sigma_1 \in [0, q-1]$ 
3:    $e \leftarrow [-\lfloor(\alpha-1)/2\rfloor, \lfloor\alpha/2\rfloor]$ 
4:    $\sigma_A = (\alpha\sigma_1 + e) \bmod q'$ 
5:    $k_1 = \lfloor\sigma_A/\beta\rfloor \in \mathbb{Z}_m$ 
6:    $v' = \sigma_A \bmod \beta$ 
7:    $v = \lfloor v'g/\beta \rfloor$   $\triangleright v \in \mathbb{Z}_g$ 
8:   return  $(k_1, v)$ 
9: end procedure
10: procedure REC( $(\sigma_2, v, \text{params})$ )  $\triangleright \sigma_2 \in [0, q-1]$ 
11:    $k_2 = \lfloor \alpha\sigma_2/\beta - (v + 1/2)/g \rfloor \bmod m$ 
12:   return  $k_2$ 
13: end procedure
```

The value $v + 1/2$, in line 11, estimates the exact value of $v'g/\beta$. Such an estimation can be more accurate, if one chooses to use the average value of all $v'g/\beta$'s such that $\lfloor v'g/\beta \rfloor = v$. Though such accuracy can improve the bound on correctness slightly, the formula calculating k_2 becomes more complicated.

3.2.1 Correctness and Security of OKCN

Recall that, for arbitrary positive integer $t \geq 1$ and any $x \in \mathbb{Z}$, $|x|_t = \min\{x \bmod t, t - x \bmod t\}$. Then, the following fact is direct from the definition of $|\cdot|_t$.

Fact 3.1. *For any $x, y, t, l \in \mathbb{Z}$ where $t \geq 1$ and $l \geq 0$, there exists $\theta \in \mathbb{Z}$ and $\delta \in [-l, l]$ such that $x = y + \theta t + \delta$.*

Theorem 3.2. *Suppose that the system parameters satisfy $(2d+1)m < q \left(1 - \frac{1}{g}\right)$ where $m \geq 2$ and $g \geq 2$. Then, the OKCN scheme is correct.*

Proof. Suppose $|\sigma_1 - \sigma_2|_q \leq d$. By Fact 3.1, there exist $\theta \in \mathbb{Z}$ and $\delta \in [-d, d]$ such that $\sigma_2 = \sigma_1 + \theta q + \delta$. From line 4 and 6 in Algorithm 1, we know that there is a $\theta' \in \mathbb{Z}$, such that $\alpha\sigma_1 + e + \theta'q' = \sigma_A = k_1\beta + v'$. And from the definition of α, β , we have $\alpha/\beta = m/q$. Taking these into the formula of k_2

in Rec (line 11 in Algorithm 1), we have

$$k_2 = \lfloor \alpha\sigma_2/\beta - (v + 1/2)/g \rfloor \bmod m \quad (1)$$

$$= \lfloor \alpha(\theta q + \sigma_1 + \delta)/\beta - (v + 1/2)/g \rfloor \bmod m \quad (2)$$

$$= \left\lfloor m(\theta - \theta') + \frac{1}{\beta}(k_1\beta + v' - e) + \frac{\alpha\delta}{\beta} - \frac{1}{g}(v + 1/2) \right\rfloor \bmod m \quad (3)$$

$$= \left\lfloor k_1 + \left(\frac{v'}{\beta} - \frac{v + 1/2}{g} \right) - \frac{e}{\beta} + \frac{\alpha\delta}{\beta} \right\rfloor \bmod m \quad (4)$$

Notice that $|v'/\beta - (v + 1/2)/g| = |v'g - \beta(v + 1/2)|/\beta g \leq 1/2g$. So

$$\left| \left(\frac{v'}{\beta} - \frac{v + 1/2}{g} \right) - \frac{e}{\beta} + \frac{\alpha\delta}{\beta} \right| \leq \frac{1}{2g} + \frac{\alpha}{\beta}(d + 1/2).$$

From the assumed condition $(2d + 1)m < q(1 - \frac{1}{g})$, we get that the right-hand side is strictly smaller than $1/2$; Consequently, after the rounding, $k_2 = k_1$. \square

Theorem 3.3. *OKCN is secure. Specifically, when $\sigma_1 \leftarrow \mathbb{Z}_q$, k_1 and v are independent, and k_1 is uniform over \mathbb{Z}_m , where the probability is taken over the sampling of σ_1 and the random coins used by Con.*

Proof. Recall that $q' = \text{lcm}(q, m)$, $\alpha = q'/q$, $\beta = q'/m$. We first demonstrate that σ_A is subject to uniform distribution over $\mathbb{Z}_{q'}$. Consider the map $f : \mathbb{Z}_q \times \mathbb{Z}_\alpha \rightarrow \mathbb{Z}_{q'}$; $f(\sigma, e) = (\alpha\sigma + e) \bmod q'$, where the elements in \mathbb{Z}_q and \mathbb{Z}_α are represented in the same way as specified in Algorithm 1. It is easy to check that f is an one-to-one map. Since $\sigma_1 \leftarrow \mathbb{Z}_q$ and $e \leftarrow \mathbb{Z}_\alpha$ are subject to uniform distributions, and they are independent, $\sigma_A = (\alpha\sigma_1 + e) \bmod q' = f(\sigma_1, e)$ is also subject to uniform distribution over $\mathbb{Z}_{q'}$.

In the similar way, defining $f' : \mathbb{Z}_m \times \mathbb{Z}_\beta \rightarrow \mathbb{Z}_{q'}$ such that $f'(k_1, v') = \beta k_1 + v'$, then f' is obviously a one-to-one map. From line 6 of Algorithm 1, $f'(k_1, v') = \sigma_A$. As σ_A is distributed uniformly over $\mathbb{Z}_{q'}$, (k_1, v') is uniformly distributed over $\mathbb{Z}_m \times \mathbb{Z}_\beta$, and so k_1 and v' are independent. As v only depends on v' , k_1 and v are independent. \square

3.2.2 Special Parameters, and Performance Speeding-Up

The first and the second line of Con (line 3 and 4 in Algorithm 1) play the role in transforming a uniform distribution over \mathbb{Z}_q to a uniform distribution over $\mathbb{Z}_{q'}$. If one chooses q, g, m to be power of 2, i.e., $q = 2^{\bar{q}}, g = 2^{\bar{g}}, m = 2^{\bar{m}}$ where $\bar{q}, \bar{g}, \bar{m} \in \mathbb{Z}$, then such transformation is not necessary. In this case Con and Rec can be simplified as follows:

Algorithm 2 OKCN power 2

```
1: params :  $q = 2^{\bar{q}}, g = 2^{\bar{g}}, m = 2^{\bar{m}}, d, aux = \{(\beta = q/m = 2^{\bar{q}-\bar{m}}, \gamma = \beta/g = 2^{\bar{q}-\bar{m}-\bar{g}})\}$ 
2: procedure CON( $\sigma_1, \text{params}$ )
3:    $k_1 = \lfloor \sigma_1 / \beta \rfloor$ 
4:    $v = \lfloor (\sigma_1 \bmod \beta) / \gamma \rfloor$ 
5:   return ( $k_1, v$ )
6: end procedure
7: procedure REC( $\sigma_2, v, \text{params}$ )
8:    $k_2 = \lfloor \sigma_2 / \beta - (v + 1/2) / g \rfloor \bmod m$ 
9:   return  $k_2$ 
10: end procedure
```

Since the random noise e used in calculating σ_A in Algorithm 1 is avoided, the correctness constraint on parameters can be relaxed, and we have the following corollary.

Corollary 3.1. *If q and m are power of 2, and d, g, m satisfy $2md < q \left(1 - \frac{1}{g}\right)$, then the KC scheme described in Algorithm 2 is both correct and secure.*

Proof. For correctness, as no additional noise e is added, one can take $e = 0$ into Formula 4, and then the correctness of Algorithm 2 directly follows from the proof of Theorem 3.2. For security, as a variation of the generic structure of Algorithm 1, the security of Algorithm 2 inherits from that of Algorithm 1. \square

If we take $\bar{g} + \bar{m} = \bar{q}$, Algorithm 2 can be further simplified into the variant depicted in Algorithm 3, with the constraint on parameters is further relaxed.

Corollary 3.2. *If m, g are power of 2, $q = m \cdot g$, and $2md < q$, then the KC scheme described in Algorithm 3 is correct and secure. Notice that the constraint on parameters is further simplified to $2md < q$ in this case.*

Proof. For correctness, supposing $|\sigma_1 - \sigma_2|_q \leq d$, by Fact 3.1, there exist $\theta \in \mathbb{Z}$ and $\delta \in [-d, d]$ such that $\sigma_2 = \sigma_1 + \theta q + \delta$. Taking this into line 8 of Algorithm 3, i.e., the formula computing k_2 , we have

$$\begin{aligned} k_2 &= \lfloor (\sigma_1 - v + \theta q + \delta) / g \rfloor \bmod m \\ &= (k_1 + \theta m + \lfloor \delta / g \rfloor) \bmod m. \end{aligned}$$

If $2md < q$, then $|\delta/g| \leq d/g < 1/2$, so that $k_2 = k_1 \bmod m = k_1$.

For security, as a special case of generic scheme described in Algorithm 1, the security of Algorithm 3 follows directly from that of Algorithm 1. \square

Algorithm 3 OKCN simple

```

1: params :  $q = 2^{\bar{q}}, g = 2^{\bar{g}}, m = 2^{\bar{m}}, d$ , where  $\bar{g} + \bar{m} = \bar{q}$ 
2: procedure CON( $\sigma_1$ , params)
3:    $k_1 = \left\lfloor \frac{\sigma_1}{g} \right\rfloor$ 
4:    $v = \sigma_1 \bmod g$ 
5:   return ( $k_1, v$ )
6: end procedure
7: procedure REC( $\sigma_2, v$ , params)
8:    $k_2 = \left\lfloor \frac{\sigma_2 - v}{g} \right\rfloor \bmod m$ 
9:   return  $k_2$ 
10: end procedure

```

4 Asymmetric Key Consensus with Noise

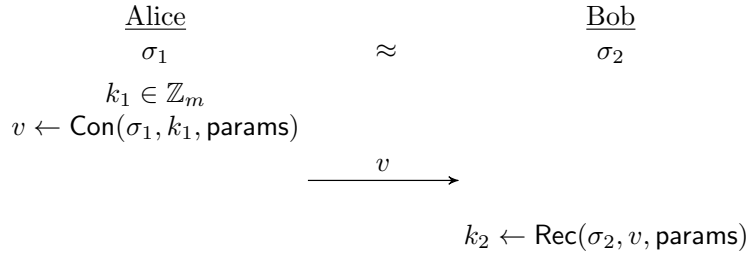


Figure 2: Brief depiction of AKC

When OKCN is used as the building tool in constructing LWE and RLWE based key exchange (KE) protocols, the party who sends the hint signal v actually plays the role of the responder. When cast into secure transport protocols in the client/server setting, e.g., the next generation of TLS 1.3 [Res], the client (corresponding to Bob) plays the role of the initiator, and the server (corresponding to Alice) plays the role of the responder. OKCN-based key exchange ensures that the shared session-key is secure, i.e., being indistinguishable from random value, but the server actually has

no essential advantage over the client in outputting the session-key. However, in some application scenarios particularly in the client/server setting, like those based upon TLS1.3, it is desirable to render the server *asymmetric* power in setting session-keys, e.g., in order to balance workloads and security or to resist the more and more vicious DDoS attacks. Another motivation is that OKCN-based key exchange, with negligible failure probability, cannot be directly transformed into a CPA-secure public-key encryption (PKE) scheme without additionally employing a CPA-secure symmetric-key encryption (SKE) scheme. These motivate us to introduce *asymmetric key consensus with noise* (AKCN), as depicted in Figure 2 and specified below.

Definition 4.1. An asymmetric key consensus scheme $AKC = (\text{params}, \text{Con}, \text{Rec})$ is specified as follows:

- $\text{params} = (q, m, g, d, \text{aux})$ denotes the system parameters, where $q, 2 \leq m, g \leq q, 1 \leq d \leq \lfloor \frac{q}{2} \rfloor$ are positive integers, and aux denotes some auxiliary values that are usually determined by (q, m, g, d) and could be set to be empty.
- $v \leftarrow \text{Con}(\sigma_1, k_1, \text{params})$: On input of $(\sigma_1 \in \mathbb{Z}_q, k_1 \in \mathbb{Z}_m, \text{params})$, the probabilistic polynomial-time conciliation algorithm Con outputs the public hint signal $v \in \mathbb{Z}_g$.
- $k_2 \leftarrow \text{Rec}(\sigma_2, v, \text{params})$: On input of $(\sigma_2, v, \text{params})$, the deterministic polynomial-time algorithm Rec outputs $k_2 \in \mathbb{Z}_m$.

Correctness: An AKC scheme is correct, if it holds $k_1 = k_2$ for any $\sigma_1, \sigma_2 \in \mathbb{Z}_q$ such that $|\sigma_1 - \sigma_2|_q \leq d$.

Security: An AKC scheme is secure, if v is independent of k_1 whenever σ_1 is uniformly distributed over \mathbb{Z}_q . Specifically, for arbitrary $\tilde{v} \in \mathbb{Z}_g$ and arbitrary $\tilde{k}_1, \tilde{k}'_1 \in \mathbb{Z}_m$, it holds that $\Pr[v = \tilde{v} | k_1 = \tilde{k}_1] = \Pr[v = \tilde{v} | k_1 = \tilde{k}'_1]$, where the probability is taken over $\sigma_1 \leftarrow \mathbb{Z}_q$ and the random coins used by Con .

When AKC is used as a building tool for key exchange, k_1 is taken uniformly at random from \mathbb{Z}_q . However, when AKC is used for public-key encryption, k_1 can be arbitrary value from the space of plaintext messages.

4.1 Efficiency Upper Bound of AKC

Similar to KC, the following theorem divulges bounds on bandwidth (parameterized by g), consensus range (parameterized by m), and correctness

(parameterized by d) for any AKC scheme. This explicit bound allows us to take balance on these parameters according to different goals or priorities among security, computational efficiency and bandwidth consumption.

Theorem 4.1. *Let $AKC = (\text{params}, \text{Con}, \text{Rec})$ be an asymmetric key consensus scheme. If AKC is correct and secure, and the parameters d, m are tight (as defined in Theorem 3.1, page 9), then it holds:*

$$2md \leq q \left(1 - \frac{m}{g}\right).$$

The proof of Theorem 4.1 is very similar to that of Theorem 3.1. Before proving Theorem 4.1, we first adjust Proposition 3.2 to the AKC setting, as following.

Proposition 4.1. *Given $\text{params} = (q, m, g, d, \text{aux})$ for an AKC scheme, then there exists $v_0 \in \mathbb{Z}_g$ such that $|S_{v_0}| \geq mq/g$.*

Proof. If k_1 is taken uniformly at random from \mathbb{Z}_m , AKC can be considered as a special KC scheme by treating $v \leftarrow \text{Con}(\sigma_1, k_1, \text{params})$ as $(k_1, v) \leftarrow \text{Con}(\sigma_1, \text{params})$. Consequently, Proposition 3.1 holds for this case. Denote $S'_v \triangleq \{(\sigma_1, k_1) \mid \Pr[v' \leftarrow \text{Con}(\sigma_1, k_1, \text{params}) : v' = v] > 0\}$. Then, S_v defined in Proposition 3.2 equals to the set containing all the values of σ_1 appeared in $(\sigma_1, \cdot) \in S'_v$. We run $\text{Con}(\sigma_1, k_1, \text{params})$ for each pair of $(\sigma_1, k_1) \in \mathbb{Z}_q \times \mathbb{Z}_m$. By the pigeonhole principle, there must exist a $v_0 \in \mathbb{Z}_g$ such that $|S'_{v_0}| \geq qm/g$. For any two pairs (σ_1, k_1) and (σ'_1, k'_1) in S'_{v_0} , if $\sigma_1 = \sigma'_1$, from Proposition 3.1 we derive that $k_1 = k'_1$, and then $(\sigma_1, k_1) = (\sigma'_1, k'_1)$. Hence, if (σ_1, k_1) and (σ'_1, k'_1) are different, then $\sigma_1 \neq \sigma'_1$, and so $|S_{v_0}| = |S'_{v_0}| \geq mq/g$. \square

Proof of Theorem 4.1. By viewing AKC, with $k_1 \leftarrow \mathbb{Z}_q$, as a special KC scheme, all the reasoning in the proof of Theorem 3.1 holds true now. At the end of the proof of Theorem 3.1, we derive $|S_{v_0}| + 2md \leq q$. By taking $|S_{v_0}| \geq mq/g$ according to Proposition 4.1, the proof is finished. \square

Comparing the formula $2md \leq q(1 - m/g)$ in Theorem 4.1 with the formula $2md \leq q(1 - 1/g)$ in Theorem 3.1, we see that the only difference is a factor m in g . This indicates that, on the same values of (q, m, d) , an AKC scheme has to use a bigger bandwidth parameter g compared to KC.

4.2 Construction and Analysis of AKCN

Algorithm 4 AKCN: Asymmetric KC with Noise

```

1: params =  $(q, m, g, d, aux)$ , where  $aux = \emptyset$ .
2: procedure CON( $\sigma_1, k_1, \text{params}$ )  $\triangleright \sigma_1 \in [0, q - 1]$ 
3:    $v = \lfloor g(\sigma_1 + \lfloor k_1 q / m \rfloor) / q \rfloor \bmod g$ 
4:   return  $v$ 
5: end procedure
6: procedure REC( $\sigma_2, v, \text{params}$ )  $\triangleright \sigma_2 \in [0, q - 1]$ 
7:    $k_2 = \lfloor m(v/g - \sigma_2/q) \rfloor \bmod m$ 
8:   return  $k_2$ 
9: end procedure

```

The AKC scheme, referred to as asymmetric KC with noise (AKCN), is depicted in Algorithm 4. We note that, in some sense, AKCN could be viewed as the generalization and optimization of the consensus mechanism proposed in [LPR13a] for CPA-secure public-key encryption. For AKCN, we can offline compute and store k_1 and $g \lfloor k_1 q / m \rfloor$ in order to accelerate online performance.

We note that the underlying AKC mechanism in the spLWE/spLWR based PKE scheme analyzed in [CKLS16] is actually an instantiation of the above AKCN scheme for the special case of $m|g|q$, where g (resp., m) in AKCN corresponds to p (resp., t) in [CKLS16].

4.2.1 Correctness and Security of AKCN

Theorem 4.2. *Suppose the parameters of AKCN satisfy $(2d + 1)m < q \left(1 - \frac{m}{g}\right)$. Then, the AKCN scheme described in Algorithm 4 is correct.*

Proof. From the formula generating v , we know that there exist $\epsilon_1, \epsilon_2 \in \mathbb{R}$ and $\theta \in \mathbb{Z}$, where $|\epsilon_1| \leq 1/2$ and $|\epsilon_2| \leq 1/2$, such that

$$v = \frac{g}{q} \left(\sigma_1 + \left(\frac{k_1 q}{m} + \epsilon_1 \right) \right) + \epsilon_2 + \theta g$$

Taking this into the formula computing k_2 in Rec, we have

$$\begin{aligned} k_2 &= \lfloor m(v/g - \sigma_2/q) \rfloor \bmod m \\ &= \left\lfloor m \left(\frac{1}{q} (\sigma_1 + k_1 q / m + \epsilon_1) + \frac{\epsilon_2}{g} + \theta - \frac{\sigma_2}{q} \right) \right\rfloor \bmod m \end{aligned}$$

$$= \left\lfloor k_1 + \frac{m}{q}(\sigma_1 - \sigma_2) + \frac{m}{q}\epsilon_1 + \frac{m}{g}\epsilon_2 \right\rfloor \bmod m$$

By Fact 3.1 (page 12), there exist $\theta' \in \mathbb{Z}$ and $\delta \in [-d, d]$ such that $\sigma_1 = \sigma_2 + \theta'q + \delta$. Hence,

$$k_2 = \left\lfloor k_1 + \frac{m}{q}\delta + \frac{m}{q}\epsilon_1 + \frac{m}{g}\epsilon_2 \right\rfloor \bmod m$$

Since $|m\delta/q + m\epsilon_1/q + m\epsilon_2/g| \leq md/q + m/2q + m/2g < 1/2$, $k_1 = k_2$. \square

Theorem 4.3. *The AKCN scheme is secure. Specifically, v is independent of k_1 when $\sigma_1 \leftarrow \mathbb{Z}_q$.*

Proof. For arbitrary $\tilde{v} \in \mathbb{Z}_g$ and arbitrary $\tilde{k}_1, \tilde{k}'_1 \in \mathbb{Z}_m$, we prove that $\Pr[v = \tilde{v} | k_1 = \tilde{k}_1] = \Pr[v = \tilde{v} | k_1 = \tilde{k}'_1]$ when $\sigma_1 \leftarrow \mathbb{Z}_q$.

For any (\tilde{k}, \tilde{v}) in $\mathbb{Z}_m \times \mathbb{Z}_g$, the event $(v = \tilde{v} \mid k_1 = \tilde{k})$ is equivalent to the event that there exists $\sigma_1 \in \mathbb{Z}_q$ such that $\tilde{v} = \lfloor g(\sigma_1 + \lfloor \tilde{k}q/m \rfloor)/q \rfloor \bmod g$. Note that $\sigma_1 \in \mathbb{Z}_q$ satisfies $\tilde{v} = \lfloor g(\sigma_1 + \lfloor \tilde{k}q/m \rfloor)/q \rfloor \bmod g$, if and only if there exist $\epsilon \in (-1/2, 1/2]$ and $\theta \in \mathbb{Z}$ such that $\tilde{v} = g(\sigma_1 + \lfloor \tilde{k}q/m \rfloor)/q + \epsilon - \theta g$. That is, $\sigma_1 = (q(\tilde{v} - \epsilon)/g - \lfloor \tilde{k}q/m \rfloor) \bmod q$, for some $\epsilon \in (-1/2, 1/2]$. Let $\Sigma(\tilde{v}, \tilde{k}) = \{\sigma_1 \in \mathbb{Z}_q \mid \exists \epsilon \in (-1/2, 1/2] \text{ s.t. } \sigma_1 = \lfloor q(\tilde{v} - \epsilon)/g - \lfloor \tilde{k}q/m \rfloor \rfloor \bmod q\}$. Defining the map $\phi : \Sigma(\tilde{v}, 0) \rightarrow \Sigma(\tilde{v}, \tilde{k})$, by setting $\phi(x) = (x - \lfloor \tilde{k}q/m \rfloor) \bmod q$. Then ϕ is obviously a one-to-one map. Hence, the cardinality of $\Sigma(\tilde{v}, \tilde{k})$ is irrelevant to \tilde{k} . Specifically, for arbitrary $\tilde{v} \in \mathbb{Z}_g$ and arbitrary $\tilde{k}_1, \tilde{k}'_1 \in \mathbb{Z}_m$, it holds that $|\Sigma(\tilde{v}, \tilde{k}_1)| = |\Sigma(\tilde{v}, \tilde{k}'_1)| = |\Sigma(\tilde{v}, 0)|$

Now, for arbitrary $\tilde{v} \in \mathbb{Z}_g$ and arbitrary $\tilde{k} \in \mathbb{Z}_m$, when $\sigma_1 \leftarrow \mathbb{Z}_q$ we have that $\Pr[v = \tilde{v} \mid k_1 = \tilde{k}] = \Pr[\sigma_1 \in \Sigma(\tilde{v}, \tilde{k}) \mid k_1 = \tilde{k}] = |\Sigma(\tilde{v}, \tilde{k})|/q = |\Sigma(\tilde{v}, 0)|/q$. The right hand side only depends on \tilde{v} , and so v is independent of k_1 . \square

4.2.2 Special Parameters and Performance Speeding-Up

We consider the parameters $q = g = 2^{\bar{q}}, m = 2^{\bar{m}}$ for positive integers \bar{q}, \bar{m} . Then the two rounding operations in line 3 of Con (in Algorithm 4) can be directly eliminated, since only integers are involved in the computation. We have the following variant described in Algorithm 5. Note that, in Algorithm 5, the modular and multiplication/division operations can be implemented by simple bitwise operations.

Algorithm 5 AKCN power 2

```
1: params :  $q = g = 2^{\bar{q}}, m = 2^{\bar{m}}, aux = \{G = q/m\}$ 
2: procedure CON( $\sigma_1, k_1, \mathbf{params}$ )
3:    $v = (\sigma_1 + k_1 \cdot G) \bmod q$ , where  $k_1 \cdot G$  can be offline computed
4:   return  $v$ 
5: end procedure
6: procedure REC( $\sigma_2, v, \mathbf{params}$ )
7:    $k_2 = \lfloor (v - \sigma_2)/G \rfloor \bmod m$ 
8:   return  $k_2$ 
9: end procedure
```

For the protocol variant presented in Algorithm 5, its correctness and security can be proved with a relaxed constraint on the parameters of (q, d, m) , as shown in the following corollary.

Corollary 4.1. *If q and m are power of 2, and d, m and q satisfy $2md < q$, then the AKCN scheme described in Algorithm 5 is correct and secure.*

Proof. For correctness, suppose $|\sigma_1 - \sigma_2|_q \leq d$, then there exist $\delta \in [-d, d]$ and $\theta \in \mathbb{Z}$ such that $\sigma_2 = \sigma_1 + \theta q + \delta$. From the formula calculating v , there exists $\theta' \in \mathbb{Z}$ such that $v = \sigma_1 + k_1 2^{\bar{q}-\bar{m}} + \theta' q$. Taking these into the formula computing k_2 , line 7 of Rec in Algorithm 5, we have

$$\begin{aligned} k_2 &= \lfloor (v - \sigma_1 - \delta - \theta q) / 2^{\bar{q}-\bar{m}} \rfloor \bmod m \\ &= \lfloor (k_1 2^{\bar{q}-\bar{m}} - \delta) / 2^{\bar{q}-\bar{m}} \rfloor \bmod m \\ &= (k_1 - \lfloor \delta / 2^{\bar{q}-\bar{m}} \rfloor) \bmod m \end{aligned}$$

If $2md < q$, then $|\delta / 2^{\bar{q}-\bar{m}}| < 1/2$, so that $k_1 = k_2$.

For security, as a special case of the generic AKCN scheme in Algorithm 4, the security of the AKCN scheme in Algorithm 5 directly follows from that of Algorithm 4. \square

5 LWE-Based Key Exchange from KC and AKC

In this section, following the protocol structure in [Pei14, ADPS16, BCD⁺16], we present the applications of OKCN and AKCN to key exchange protocols based on LWE. Denote by $(\lambda, n, q, \chi, KC, l_A, l_B, t)$ the underlying parameters, where λ is the security parameter, $q \geq 2$, n , l_A and l_B are positive integers that are polynomial in λ (for protocol symmetry, l_A and l_B are usually set to be equal and are actually small constant). To save bandwidth,

we cut off t least significant bits of \mathbf{Y}_2 before send it to Alice. Denote by χ a (small) noise distribution over \mathbb{Z}_q , and by **Gen** a pseudo-random generator (PRG) generating the matrix \mathbf{A} from a small seed. For presentation simplicity, we assume $\mathbf{A} \in \mathbb{Z}_q^{n \times n}$ to be square matrix. The length of random seed, i.e., κ , is typically set to be 256.

Let $KC = (\text{params}, \text{Con}, \text{Rec})$ be a *correct* and *secure* KC scheme, where **params** is set to be (q, g, m, d) . The KC-based key exchange protocol from LWE is depicted in Figure 3, and the actual session-key is derived from \mathbf{K}_1 and \mathbf{K}_2 via some key derivation function KDF that may be modelled as a random oracle. There, for presentation simplicity, the **Con** and **Rec** functions are applied to matrices, meaning they are applied to each of the coordinates separately. Note that $2^t \mathbf{Y}'_2 + 2^{t-1} \mathbf{1}$ is an approximation of \mathbf{Y}_2 , so we have $\Sigma_1 \approx \mathbf{X}_1^T \mathbf{Y}_2 = \mathbf{X}_1^T \mathbf{A}^T \mathbf{X}_2 + \mathbf{X}_1^T \mathbf{E}_2$, $\Sigma_2 = \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma = \mathbf{X}_1^T \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$. As we choose $\mathbf{X}_1, \mathbf{X}_2, \mathbf{E}_1, \mathbf{E}_2$ according to a small noise distribution χ , the main part of Σ_1 and that of Σ_2 are the same $\mathbf{X}_1^T \mathbf{A}^T \mathbf{X}_2$. Hence, the corresponding coordinates of Σ_1 and Σ_2 are close in the sense of $|\cdot|_q$, from which some key consensus can be reached. The failure probability depends upon the number of bits we cut t , underlying distribution χ and the distance parameter d , which will be analyzed in detail in subsequent sections.

For presentation simplicity, we have described the LWE-based key exchange protocol from a KC scheme. But it can be trivially adapted to work on any correct and secure AKC scheme. In this case, the responder user Bob simply chooses $\mathbf{K}_2 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$, and the output of $\text{Con}(\Sigma_2, \mathbf{K}_2, \text{params})$ is simply defined to be $(\mathbf{K}_2, \mathbf{V})$. For presentation simplicity, in the following security definition and analysis we also simply assume that the output of the PRG **Gen** is truly random.

On operations related to Matrix \mathbf{A} . Firstly, in this work, as the dimension of the matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times n}$ in our scheme is not large enough, we use the naïve matrix multiplication. But we remark that matrix operations can be asymptotically speeded up with the algorithms developed in [CW90, Str69].

Secondly, for LWE-based schemes when $q \leq 2^{16}$ is power of 2, we use a 16-bit unsigned integer to store an element of the matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times n}$. Since our q is a divisor of 2^{16} , only one bitwise AND-operation is needed to modulo q .

Thirdly, when building a PKE scheme from KC-based key exchange protocol from LWE, the matrix \mathbf{A} has to be fixed and public, which dominates the size of public key. But if we are only concerned with key exchange, the matrix can be generated by a small random seed via a PRG, which, on the one hand, can much save communication bandwidth, and on the other hand can resist the all-for-the-price-of-one attack [BCD⁺16]. For the case that

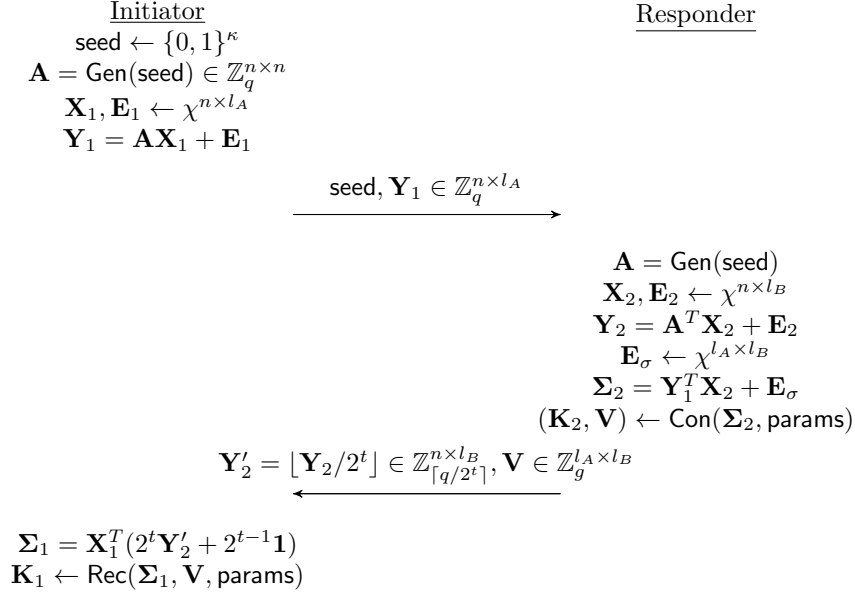


Figure 3: KC-based key exchange from LWE, where $\mathbf{K}_1, \mathbf{K}_2 \in \mathbb{Z}_m^{l_A \times l_B}$ and $|\mathbf{K}_1| = |\mathbf{K}_2| = l_A l_B |m|$. $\mathbf{1}$ refers to the matrix which every elements are 1.

q is a power of 2, sampling a number range from 0 to $q - 1$ randomly can be done by a simple bitwise AND-operation. As for the other forms of q , a naïve method is to accept a random number if it is less than q , and reject it if it is not and then use the next random number, and so forth. The work [GS16] proposes a sample method with lower rejection probability and fewer random numbers.

Finally, in case the memory of a client device is limited and inconvenient for saving the whole matrix \mathbf{A} , a solution is to generate, use and disposal \mathbf{A} on-the-fly [BCD⁺16]. Specifically, Initiator (corresponding to the client in TLS1.3) can generate one row or one column of \mathbf{A} one time, depending on which side \mathbf{A} is in the multiplication.

5.1 Security Analysis

Definition 5.1. A KC or AKC based key exchange protocol from LWE is secure, if for any sufficiently large security parameter λ and any PPT adversary \mathcal{A} , $|\Pr[b' = b] - \frac{1}{2}|$ is negligible, as defined w.r.t. game G_0 specified in Algorithm 6.

Algorithm 6 Game G_0

```
1:  $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$ 
2:  $\mathbf{X}_1, \mathbf{E}_1 \leftarrow \chi^{n \times l_A}$ 
3:  $\mathbf{Y}_1 = \mathbf{A}\mathbf{X}_1 + \mathbf{E}_1$ 
4:  $\mathbf{X}_2, \mathbf{E}_2 \leftarrow \chi^{n \times l_B}$ 
5:  $\mathbf{Y}_2 = \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2$ 
6:  $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$ 
7:  $\Sigma_2 = \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$ 
8:  $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\Sigma_2, \text{params})$ 
9:  $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$ 
10:  $b \leftarrow \{0, 1\}$ 
11:  $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$ 
```

Before starting to prove the security, we first recall some basic properties of the LWE assumption. The following lemma is derived by a direct hybrid argument [PVW08, BCD⁺16].

Lemma 5.1 (LWE in the matrix form). *For positive integer parameters $(\lambda, n, q \geq 2, l, t)$, where n, q, l, t all are polynomial in λ , and a distribution χ over \mathbb{Z}_q , denote by $L_\chi^{(l,t)}$ the distribution over $\mathbb{Z}_q^{t \times n} \times \mathbb{Z}_q^{t \times l}$ generated by taking $\mathbf{A} \leftarrow \mathbb{Z}_q^{t \times n}, \mathbf{S} \leftarrow \chi^{n \times l}, \mathbf{E} \leftarrow \chi^{t \times l}$ and outputting $(\mathbf{A}, \mathbf{A}\mathbf{S} + \mathbf{E})$. Then, under the standard LWE assumption on indistinguishability between $A_{q,s,\chi}$ (with $\mathbf{s} \leftarrow \chi^n$) and $\mathcal{U}(\mathbb{Z}_q^n \times \mathbb{Z}_q)$, no PPT distinguisher \mathcal{D} can distinguish, with non-negligible probability, between the distribution $L_\chi^{(l,t)}$ and $\mathcal{U}(\mathbb{Z}_q^{t \times n} \times \mathbb{Z}_q^{t \times l})$ for sufficiently large λ .*

Theorem 5.1. *If $(\text{params}, \text{Con}, \text{Rec})$ is a correct and secure KC or AKC scheme, the key exchange protocol described in Figure 3 is secure under the (matrix form of) LWE assumption.*

Proof. The proof is similar to, but actually simpler than, that in [Pei14, BCD⁺16]. The general idea is that we construct a sequence of games: G_0 , G_1 and G_2 , where G_0 is the original game for defining security. In every move from game G_i to G_{i+1} , $0 \leq i \leq 1$, we change a little. All games G_i 's share the same PPT adversary \mathcal{A} , whose goal is to distinguish between the matrices chosen uniformly at random and the matrices generated in the actual key exchange protocol. Denote by T_i , $0 \leq i \leq 2$, the event that $b = b'$ in Game G_i . Our goal is to prove that $\Pr[T_0] < 1/2 + \text{negl}$, where negl is a negligible function in λ . For ease of readability, we re-produce game G_0 below. For presentation simplicity, in the subsequent analysis, we always

assume the underlying KC or AKC is correct. The proof can be trivially extended to the case that correctness holds with overwhelming probability (i.e., failure occurs with negligible probability).

Algorithm 7 Game G_0	Algorithm 8 Game G_1
1: $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$	1: $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$
2: $\mathbf{X}_1, \mathbf{E}_1 \leftarrow \chi^{n \times l_A}$	2: $\mathbf{X}_1, \mathbf{E}_1 \leftarrow \chi^{n \times l_A}$
3: $\mathbf{Y}_1 = \mathbf{A}\mathbf{X}_1 + \mathbf{E}_1$	3: $\mathbf{Y}_1 \leftarrow \mathbb{Z}_q^{n \times l_A}$
4: $\mathbf{X}_2, \mathbf{E}_2 \leftarrow \chi^{n \times l_B}$	4: $\mathbf{X}_2, \mathbf{E}_2 \leftarrow \chi^{n \times l_B}$
5: $\mathbf{Y}_2 = \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2$	5: $\mathbf{Y}_2 = \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2$
6: $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$	6: $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$
7: $\Sigma_2 = \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$	7: $\Sigma_2 = \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$
8: $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\Sigma_2, \text{params})$	8: $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\Sigma_2, \text{params})$
9: $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$	9: $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$
10: $b \leftarrow \{0, 1\}$	10: $b \leftarrow \{0, 1\}$
11: $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$	11: $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$

Lemma 5.2. $|\Pr[T_0] - \Pr[T_1]| < \text{negl}$, under the indistinguishability between $L_\chi^{(l_A, n)}$ and $\mathcal{U}(\mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^{n \times l_A})$.

Proof. Construct a distinguisher \mathcal{D} , in Algorithm 9, who tries to distinguish $L_\chi^{(l_A, n)}$ from $\mathcal{U}(\mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^{n \times l_A})$.

Algorithm 9 Distinguisher \mathcal{D}	
1: procedure $\mathcal{D}(\mathbf{A}, \mathbf{B})$	$\triangleright \mathbf{A} \in \mathbb{Z}_q^{n \times n}, \mathbf{B} \in \mathbb{Z}_q^{n \times l_A}$
2: $\mathbf{Y}_1 = \mathbf{B}$	
3: $\mathbf{X}_2, \mathbf{E}_2 \leftarrow \chi^{n \times l_B}$	
4: $\mathbf{Y}_2 = \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2$	
5: $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$	
6: $\Sigma_2 = \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$	
7: $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\Sigma_2, \text{params})$	
8: $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$	
9: $b \leftarrow \{0, 1\}$	
10: $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$	
11: if $b' = b$ then	
12: return 1	
13: else	
14: return 0	
15: end if	
16: end procedure	

If (\mathbf{A}, \mathbf{B}) is subject to $L_\chi^{(l_A, n)}$, then \mathcal{D} perfectly simulates G_0 . Hence, $\Pr[\mathcal{D}(L_\chi^{(l_A, n)}) = 1] = \Pr[T_0]$. On the other hand, if (\mathbf{A}, \mathbf{B}) is chosen uniformly at random from $\mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^{n \times l_A}$, which are denoted as $(\mathbf{A}^\mathcal{U}, \mathbf{B}^\mathcal{U})$, then \mathcal{D} perfectly simulates G_1 . So, $\Pr[\mathcal{D}(\mathbf{A}^\mathcal{U}, \mathbf{B}^\mathcal{U}) = 1] = \Pr[T_1]$. Hence, $|\Pr[T_0] - \Pr[T_1]| = |\Pr[\mathcal{D}(L_\chi^{(l_A, n)}) = 1] - \Pr[\mathcal{D}(\mathbf{A}^\mathcal{U}, \mathbf{B}^\mathcal{U}) = 1]| < \text{negl}$. \square

Algorithm 10 Game G_1	Algorithm 11 Game G_2
1: $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$	1: $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$
2: $\mathbf{X}_1, \mathbf{E}_1 \leftarrow \chi^{n \times l_A}$	2: $\mathbf{X}_1, \mathbf{E}_1 \leftarrow \chi^{n \times l_A}$
3: $\mathbf{Y}_1 \leftarrow \mathbb{Z}_q^{n \times l_A}$	3: $\mathbf{Y}_1 \leftarrow \mathbb{Z}_q^{n \times l_A}$
4: $\mathbf{X}_2, \mathbf{E}_2 \leftarrow \chi^{n \times l_B}$	4: $\mathbf{X}_2, \mathbf{E}_2 \leftarrow \chi^{n \times l_B}$
5: $\mathbf{Y}_2 = \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2$	5: $\mathbf{Y}_2 \leftarrow \mathbb{Z}_q^{n \times l_B}$
6: $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$	6: $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$
7: $\Sigma_2 = \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$	7: $\Sigma_2 \leftarrow \mathbb{Z}_q^{l_A \times l_B}$
8: $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\Sigma_2, \text{params})$	8: $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\Sigma_2, \text{params})$
9: $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$	9: $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$
10: $b \leftarrow \{0, 1\}$	10: $b \leftarrow \{0, 1\}$
11: $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$	11: $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$

Lemma 5.3. $|\Pr[T_1] - \Pr[T_2]| < \text{negl}$, under the indistinguishability between $L_\chi^{(l_B, n+l_A)}$ and $\mathcal{U}(\mathbb{Z}_q^{(n+l_A) \times n} \times \mathbb{Z}_q^{(n+l_A) \times l_B})$.

Proof. As \mathbf{Y}_1 is subject to uniform distribution in G_1 , $(\mathbf{Y}_1^T, \Sigma_2)$ can be regarded as an $L_\chi^{(l_B, l_A)}$ sample of secret \mathbf{X}_2 and noise \mathbf{E}_σ . Based on this observation, we construct the following distinguisher \mathcal{D}' .

Algorithm 12 Distinguisher \mathcal{D}'

```

1: procedure  $\mathcal{D}'(\mathbf{A}', \mathbf{B})$  where  $\mathbf{A}' \in \mathbb{Z}_q^{(n+l_A) \times n}$ ,  $\mathbf{B} \in \mathbb{Z}_q^{(n+l_A) \times l_B}$ 
2:   Denote  $\mathbf{A}' = \begin{pmatrix} \mathbf{A}^T \\ \mathbf{Y}_1^T \end{pmatrix}$   $\triangleright \mathbf{A} \in \mathbb{Z}_q^{n \times n}, \mathbf{Y}_1^T \in \mathbb{Z}_q^{l_A \times n}$ 
3:   Denote  $\mathbf{B} = \begin{pmatrix} \mathbf{Y}_2 \\ \mathbf{\Sigma}_2 \end{pmatrix}$   $\triangleright \mathbf{Y}_2 \in \mathbb{Z}_q^{n \times l_B}, \mathbf{\Sigma}_2 \in \mathbb{Z}_q^{l_A \times l_B}$ 
4:    $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\mathbf{\Sigma}_2, \text{params})$ 
5:    $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$ 
6:    $b \leftarrow \{0, 1\}$ 
7:    $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$ 
8:   if  $b' = b$  then
9:     return 1
10:  else
11:    return 0
12:  end if
13: end procedure

```

If $(\mathbf{A}', \mathbf{B})$ is subject to $L_\chi^{(l_B, n+l_A)}$, $\mathbf{A}' \leftarrow \mathbb{Z}_q^{(n+l_A) \times n}$ corresponds to $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$ and $\mathbf{Y}_1 \leftarrow \mathbb{Z}_q^{n \times l_A}$ in G_1 ; and $\mathbf{S} \leftarrow \chi^{n \times l_B}$ (resp., $\mathbf{E} \leftarrow \chi^{(n+l_A) \times l_B}$) in generating $(\mathbf{A}', \mathbf{B})$ corresponds to $\mathbf{X}_2 \leftarrow \chi^{n \times l_B}$ (resp., $\mathbf{E}_2 \leftarrow \chi^{n \times l_B}$ and $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$) in G_1 . In this case, we have

$$\begin{aligned}
\mathbf{B} &= \mathbf{A}'\mathbf{S} + \mathbf{E} = \begin{pmatrix} \mathbf{A}^T \\ \mathbf{Y}_1^T \end{pmatrix} \mathbf{X}_2 + \begin{pmatrix} \mathbf{E}_2 \\ \mathbf{E}_\sigma \end{pmatrix} \\
&= \begin{pmatrix} \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2 \\ \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma \end{pmatrix} = \begin{pmatrix} \mathbf{Y}_2 \\ \mathbf{\Sigma}_2 \end{pmatrix}
\end{aligned}$$

Hence $\Pr[\mathcal{D}'(L_\chi^{(l_B, n+l_A)}) = 1] = \Pr[T_1]$.

On the other hand, if $(\mathbf{A}', \mathbf{B})$ is subject to uniform distribution $\mathcal{U}(\mathbb{Z}_q^{(n+l_A) \times n} \times \mathbb{Z}_q^{(n+l_A) \times l_B})$, then $\mathbf{A}, \mathbf{Y}_1, \mathbf{Y}_2, \mathbf{\Sigma}_2$ all are also uniformly random; So, the view of \mathcal{D}' in this case is the same as that in game G_2 . Hence, $\Pr[\mathcal{D}'(\mathbf{A}', \mathbf{B}) = 1] = \Pr[T_2]$ in this case. Then $|\Pr[T_1] - \Pr[T_2]| = |\Pr[\mathcal{D}'(L_\chi^{(l_B, n+l_A)}) = 1] - \Pr[\mathcal{D}'(\mathcal{U}(\mathbb{Z}_q^{(n+l_A) \times n} \times \mathbb{Z}_q^{(n+l_A) \times l_B})) = 1]| < \text{negl}$. \square

Lemma 5.4. *If the underlying KC or AKC is secure, $\Pr[T_2] = \frac{1}{2}$.*

Proof. Note that, in Game G_2 , for any $1 \leq i \leq l_A$ and $1 \leq j \leq l_B$, $(\mathbf{K}_2^0[i, j], \mathbf{V}[i, j])$ only depends on $\mathbf{\Sigma}_2[i, j]$, and $\mathbf{\Sigma}_2$ is subject to uniform distribution. By the *security* of KC, we have that, for each pair (i, j) , $\mathbf{K}_2^0[i, j]$

and $\mathbf{V}[i, j]$ are independent, and $\mathbf{K}_2^0[i, j]$ is uniform distributed. Hence, \mathbf{K}_2^0 and \mathbf{V} are independent, and \mathbf{K}_2^0 is uniformly distributed, which implies that $\Pr[T_2] = 1/2$. \square

This finishes the proof of Theorem 5.1. \square

Theorem 5.1 indicates that, achieving *secure* LWE-based key exchange is reduced to construct *correct* and *secure* KC or AKC schemes. In the above security analysis, we have assumed that no failure occurs or failure occurs with only negligible probability. Here, failure means that there exists (i, j) , $1 \leq i \leq l_A$ and $1 \leq j \leq l_B$, such that $|\Sigma_1[i, j] - \Sigma_2[i, j]|_q > d$. The correctness of the protocol depends upon the underlying error distributions, which are discussed in the next subsection.

5.2 Noise Distributions and Correctness

For a *correct* KC with parameter d , if the distance of corresponding elements of Σ_1 and Σ_2 is less than d in the sense of $|\cdot|_q$, then the scheme depicted in Figure 3 is correct. Denote $\epsilon(\mathbf{Y}_2) = 2^t \lfloor \mathbf{Y}_2 / 2^t \rfloor + 2^{t-1} \mathbf{1} - \mathbf{Y}_2$. Then

$$\begin{aligned} \Sigma_1 - \Sigma_2 &= \mathbf{X}_1^T (2^t \mathbf{Y}_2' + 2^{t-1} \mathbf{1}) - \mathbf{Y}_1^T \mathbf{X}_2 - \mathbf{E}_\sigma \\ &= \mathbf{X}_1^T (\mathbf{Y}_2 + \epsilon(\mathbf{Y}_2)) - \mathbf{Y}_1^T \mathbf{X}_2 - \mathbf{E}_\sigma \\ &= \mathbf{X}_1^T (\mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2 + \epsilon(\mathbf{Y}_2)) - (\mathbf{A} \mathbf{X}_1 + \mathbf{E}_1)^T \mathbf{X}_2 - \mathbf{E}_\sigma \\ &= \mathbf{X}_1^T (\mathbf{E}_2 + \epsilon(\mathbf{Y}_2)) - \mathbf{E}_1^T \mathbf{X}_2 - \mathbf{E}_\sigma \end{aligned}$$

We consider each pair of elements in matrix Σ_1, Σ_2 separately, then derive the overall error rate by *union bound*. Now, we only need to consider the case $l_A = l_B = 1$. In this case, $\mathbf{X}_i, \mathbf{E}_i, \mathbf{Y}_i, (i = 1, 2)$ are column vectors in \mathbb{Z}_q^n , and $\mathbf{E}_\sigma \in \mathbb{Z}_q$.

If \mathbf{Y}_2 is independent of $(\mathbf{X}_2, \mathbf{E}_2)$, then we can directly calculate the distribution of $\sigma_1 - \sigma_2$. But now \mathbf{Y}_2 depends on $(\mathbf{X}_2, \mathbf{E}_2)$. To overcome this difficulty, we show that \mathbf{Y}_2 is independent of $(\mathbf{X}_2, \mathbf{E}_2)$ under a condition of \mathbf{X}_2 that happens with very high probability.

Theorem 5.2. *For any positive integer q, n , and a column vector $\mathbf{s} \in \mathbb{Z}_q^n$, let $\phi_{\mathbf{s}}$ denote map $\mathbb{Z}_q^n \rightarrow \mathbb{Z}_q : \phi_{\mathbf{s}}(\mathbf{x}) = \mathbf{x}^T \mathbf{s}$. If there exists a coordinate of \mathbf{s} which is not zero divisor in ring \mathbb{Z}_q , then map $\phi_{\mathbf{s}}$ is surjective.*

Proof. Let us assume one coordinate of \mathbf{s} , say s , has no zero divisor in ring \mathbb{Z}_q . Then the map between two \mathbb{Z}_q -modules deduced by $s: \mathbb{Z}_q \rightarrow \mathbb{Z}_q, x \mapsto sx$ is injective, and thus surjective. Hence, $\phi_{\mathbf{s}}$ is surjective. \square

For a column vector \mathbf{s} composed by random variables, denote by $F(\mathbf{s})$ the event that $\phi_{\mathbf{s}}$ is surjective. The following theorem gives a lower bound of probability of $F(\mathbf{s})$, where $\mathbf{s} \leftarrow \chi^n$. In our application, this lower bound is very close to 1.

Theorem 5.3. *Let p_0 be the probability that e is a zero divisor in ring \mathbb{Z}_q , where e is subject to χ . Then $\Pr[\mathbf{s} \leftarrow \chi^n : F(\mathbf{s})] \geq 1 - p_0^n$*

Proof. From Theorem 5.2, if $\phi_{\mathbf{s}}$ is not surjective, then all coordinates of \mathbf{s} are zero divisors. Then $\Pr[\neg \mathbf{s} \leftarrow \chi^n : F(\mathbf{s})] \leq p_0^n$, and the proof is finished. \square

Theorem 5.4. *If $\mathbf{s}, \mathbf{e} \leftarrow \chi^n$, $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$, $\mathbf{y} = \mathbf{A}\mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^n$, then under the condition $F(\mathbf{s})$, \mathbf{y} is independent of (\mathbf{s}, \mathbf{e}) , and is uniformly distributed over \mathbb{Z}_q^n .*

Proof. For all $\tilde{\mathbf{y}}, \tilde{\mathbf{s}}, \tilde{\mathbf{e}}$, $\Pr[\mathbf{y} = \tilde{\mathbf{y}} \mid \mathbf{s} = \tilde{\mathbf{s}}, \mathbf{e} = \tilde{\mathbf{e}}, F(\mathbf{s})] = \Pr[\mathbf{A}\tilde{\mathbf{s}} = \tilde{\mathbf{y}} - \tilde{\mathbf{e}} \mid \mathbf{s} = \tilde{\mathbf{s}}, \mathbf{e} = \tilde{\mathbf{e}}, F(\mathbf{s})]$. Let $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n)^T$, $\tilde{\mathbf{y}} - \tilde{\mathbf{e}} = (c_1, c_2, \dots, c_n)^T$, where $\mathbf{a}_i \in \mathbb{Z}_q^n$, and $c_i \in \mathbb{Z}_q$, for every $1 \leq i \leq n$. Since $\phi_{\mathbf{s}}$ is surjective, the number of possible choice of \mathbf{a}_i satisfying $\mathbf{a}_i^T \cdot \tilde{\mathbf{s}} = c_i$ is $|\text{Ker}\phi_{\mathbf{s}}| = q^{n-1}$. Hence, $\Pr[\mathbf{A}\tilde{\mathbf{s}} = \tilde{\mathbf{y}} - \tilde{\mathbf{e}} \mid \mathbf{s} = \tilde{\mathbf{s}}, \mathbf{e} = \tilde{\mathbf{e}}, F(\mathbf{s})] = (q^{n-1})^n / q^{n^2} = 1/q^n$. Since the right hand side is the constant $1/q^n$, the distribution of \mathbf{y} is uniform over \mathbb{Z}_q^n , and is irrelevant of (\mathbf{s}, \mathbf{e}) . \square

We now begin to analyze the error rate of the scheme presented in Figure 3.

Denote by E the event $|\mathbf{X}_1^T(\mathbf{E}_2 + \epsilon(\mathbf{Y}_2)) - \mathbf{E}_1^T \mathbf{X}_2 - \mathbf{E}_\sigma|_q > d$. Then $\Pr[E] = \Pr[E|F(\mathbf{S})] \Pr[F(\mathbf{S})] + \Pr[E|\neg F(\mathbf{S})] \Pr[\neg F(\mathbf{S})]$. From Theorem 5.4, we replace $\mathbf{Y}_2 = \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2$ in the event $E|F(\mathbf{S})$ with uniformly distributed \mathbf{Y}_2 . Then,

$$\begin{aligned} \Pr[E] &= \Pr[\mathbf{Y}_2 \leftarrow \mathbb{Z}_q^n : E|F(\mathbf{S})] \Pr[F(\mathbf{S})] + \Pr[E|\neg F(\mathbf{S})] \Pr[\neg F(\mathbf{S})] \\ &= \Pr[\mathbf{Y}_2 \leftarrow \mathbb{Z}_q^n : E|F(\mathbf{S})] \Pr[F(\mathbf{S})] + \Pr[\mathbf{Y}_2 \leftarrow \mathbb{Z}_q^n : E|\neg F(\mathbf{S})] \Pr[\neg F(\mathbf{S})] \\ &\quad + \Pr[E|\neg F(\mathbf{S})] \Pr[\neg F(\mathbf{S})] - \Pr[\mathbf{Y}_2 \leftarrow \mathbb{Z}_q^n : E|\neg F(\mathbf{S})] \Pr[\neg F(\mathbf{S})] \\ &= \Pr[\mathbf{Y}_2 \leftarrow \mathbb{Z}_q^n : E] + \epsilon \end{aligned}$$

where $|\epsilon| \leq \Pr[\neg F(\mathbf{S})]$. By Theorem 5.3, ϵ is very small in practice, because p_0 is usually far from 1, and n is very large, so we simply ignore ϵ . If \mathbf{Y}_2 is uniformly distributed, then $\epsilon(\mathbf{Y}_2)$ is a centered uniform distribution. Then distribution of $\mathbf{X}_1^T(\mathbf{E}_2 + \epsilon(\mathbf{Y}_2)) - \mathbf{E}_1^T \mathbf{X}_2 - \mathbf{E}_\sigma$ can be directly computed by programs.

As noted in [ADPS16, BCD⁺16], sampling from rounded Gaussian distribution (i.e., sampling from a discrete Gaussian distribution to a high precision) constitutes one of major efficiency bottleneck. In this work, for LWE-based key exchange, we are mainly concerned with the following two kinds of efficiently sampleable distributions.

5.2.1 Binary Distribution

Binary-LWE is a variation of LWE, where the noise distribution is set to be $\chi = \mathcal{U}(\{0, 1\})$. With respect to $m = n \cdot (1 + \Omega(1/\log n))$ samples and large enough polynomial $q \geq n^{O(1)}$, the hardness of binary-LWE is established in [MP13], with a reduction from some approximation lattice problem in dimension $\Theta(n/\log n)$. Concrete error rate can be calculated on the concrete parameters by the method in previous section.

For KC-based key exchange from binary-LWE, we have the following theorem, which means that it is correct when the underlying parameter d satisfies $d \geq n + 1$. For LWE-based KE from OKCN, where $2md < q$, we get that it is correct when $q > 2m(n + 1)$. Actually, this theorem has already been implied in the above analysis.

Theorem 5.5. *If $\chi = \mathcal{U}(\{0, 1\})$, and $(params, Con, Rec)$ is a correct KC or AKC scheme where $d \geq n + 1$, the key exchange protocol described in Algorithm 3 is correct.*

Proof. We prove that, for any (i, j) , $1 \leq i \leq l_A$ and $1 \leq j \leq l_B$, $|\Sigma_1[i, j] - \Sigma_2[i, j]|_q \leq d$ holds true. Denote $\mathbf{X}_1 = (\hat{\mathbf{X}}_1, \hat{\mathbf{X}}_2, \dots, \hat{\mathbf{X}}_{l_A})$, $\mathbf{E}_1 = (\hat{\mathbf{E}}_1, \hat{\mathbf{E}}_2, \dots, \hat{\mathbf{E}}_{l_A})$, and $\mathbf{X}_2 = (\hat{\mathbf{X}}'_1, \hat{\mathbf{X}}'_2, \dots, \hat{\mathbf{X}}'_{l_B})$, $\mathbf{E}_2 = (\hat{\mathbf{E}}'_1, \hat{\mathbf{E}}'_2, \dots, \hat{\mathbf{E}}'_{l_B})$, where $\hat{\mathbf{X}}_i, \hat{\mathbf{X}}'_i, \hat{\mathbf{E}}_i, \hat{\mathbf{E}}'_i \in \{0, 1\}^n$. Then

$$\begin{aligned} |\Sigma_1[i, j] - \Sigma_2[i, j]|_q &= \left| \hat{\mathbf{X}}_i^T \hat{\mathbf{E}}'_j - \hat{\mathbf{E}}_i^T \hat{\mathbf{X}}'_j - \mathbf{E}_\sigma[i, j] \right|_q \\ &= |\mathbf{E}_\sigma[i, j]|_q \\ &\leq n + 1 \leq d \end{aligned}$$

□

However, cautions should be taken when deploying key exchange protocols based upon binary-LWE. By noting that any Binary-LWE sample satisfies a quadric equation, if no less than n^2 samples can be used for an adversary, the secret and noise can be recovered easily. The work [AG11] proposes an algorithm for binary-LWE with time complexity $2^{\tilde{O}(\alpha q)^2}$. If $\alpha q = o(\sqrt{n})$,

this algorithm is subexponential, but it requires $2^{\tilde{O}((\alpha q)^2)}$ samples. When $m \log q / (n + m) = o(n / \log n)$, [KF15] proposes a distinguishing attack with time complexity $2^{\frac{n/2+o(n)}{\ln(m \log q / (n+m)-1)}}$, which results in a subexponential-time algorithm if m grows linearly in n .

5.2.2 Discrete distribution

It is suggested in [ADPS16, BCD⁺16] that rounded Gaussian distribution can be replaced by discrete distribution that is very close to rounded Gaussian in the sense of Rényi divergence [BLL⁺15].

Definition 5.2 ([BLL⁺15]). *For two discrete distributions P, Q satisfying $\text{Supp}(P) \subseteq \text{Supp}(Q)$, their a -order Rényi divergence, for some $a > 1$, is*

$$R_a(P||Q) = \left(\sum_{x \in \text{Supp}(P)} \frac{P(x)^a}{Q(x)^{a-1}} \right)^{\frac{1}{a-1}}.$$

Lemma 5.5 ([BLL⁺15]). *Letting $a > 1$, P and Q are two discrete distributions satisfying $\text{Supp}(P) \subseteq \text{Supp}(Q)$, then we have*

Multiplicativity: *Let P and Q be two distributions of random variable (Y_1, Y_2) . For $i \in \{1, 2\}$, let P_i and Q_i be the margin distribution of Y_i over P and Q respectively. If Y_1 and Y_2 , under P and Q respectively, are independent, then*

$$R_a(P||Q) = R_a(P_1||Q_1) \cdot R_a(P_2||Q_2).$$

Probability Preservation: *Let $A \subseteq \text{Supp}(Q)$ be an event, then*

$$Q(A) \geq P(A)^{\frac{a}{a-1}} / R_a(P||Q).$$

Note that, when the underlying key derivation function KDF is modelled as a random oracle, an attacker is considered to be successful only if it can recover the entire consensus bits. Denote by E the event that a PPT attacker can successfully and entirely recover the bits of $\mathbf{K}_1 = \mathbf{K}_2$. By Lemma 5.5, we have that $\Pr_{\text{rounded Gaussian}}[E] > \Pr_{\text{discrete}}[E]^{a/(a-1)} / R_a^{n \cdot (l_A + l_B) + l_A \cdot l_B}(\chi || \bar{\phi})$, where $\bar{\phi}$ is the rounded Gaussian distribution, and χ is the discrete distribution.

In this work, for LWE-based key exchange, we use the following two classes of discrete distributions, which are specified in Table 1 and Table 2

dist.	bits	var.	probability of							order	divergence
			0	± 1	± 2	± 3	± 4	± 5			
D_1	8	1.10	94	62	17	2			15.0	1.0015832	
D_2	12	0.90	1646	992	216	17			75.0	1.0003146	
D_3	12	1.66	1238	929	393	94	12	1	30.0	1.0002034	
D_4	16	1.66	19794	14865	6292	1499	200	15	500.0	1.0000274	

Table 1: Discrete distributions proposed in this work, and their Rényi divergences.

dist.	bits	var.	probability of							order	divergence
			0	± 1	± 2	± 3	± 4	± 5	± 6		
\bar{D}_1	8	1.25	88	61	20	3				25.0	1.0021674
\bar{D}_2	12	1.00	1570	990	248	24	1			40.0	1.0001925
\bar{D}_3	12	1.75	1206	919	406	104	15	1		100.0	1.0003011
\bar{D}_4	16	1.75	19304	14700	6490	1659	245	21	1	500.0	1.0000146

Table 2: Discrete distributions for Frodo [BCD⁺16], and their Rényi divergences

respectively, where “bits” refers to the number of bits required to sample the distribution and “var.” means the standard variation of the Gaussian distribution approximated. We remark that the discrete distributions specified in Table 2 are just those specified and used in [BCD⁺16] for the LWE-based Frodo scheme.

5.3 Instantiations, and Comparisons with Frodo

In this section, we present several instantiations of our LWE-based key exchange protocol proposed in Algorithm 3, with carefully chosen and evaluated parameters, and make a detailed comparison with the protocol Frodo proposed in [BCD⁺16].

- Above all, OKCN and AKCN are developed within a more generalized framework, while Frodo is proposed on parameters of special kinds (in particular, $g = 2$) and does not consider the technique of cutting off some least significant bits.
- Frodo is based on *symmetric* key consensus mechanism, while our LWE-based KE can be based on either OKCN or AKCN. When instantiated with AKCN, our key exchange protocol allows online/offline

and parallel computations; In particular, the session-key can be pre-determined by the responder (i.e., the server when our KE protocol is used within TLS1.3) and thus can be used to encrypt messages even before the run of KE protocol starts. This feature is particularly helpful for the server to resist DDoS attacks or to balance workloads and security.

- Both OKCN and AKCN can choose tighter parameters than Frodo. For example, for both “OKCN simple” proposed in Algorithm 3 and “AKCN power 2” proposed in Algorithm 5, they achieve a tight parameter constraint, specifically, $2md < q$. In comparison, the parameter constraint achieved by Frodo is $4md < q$. As we shall see, such a difference is one source that allows us to achieve better trade-offs among error rate, security, (computational and bandwidth) efficiency, and consensus range. In particular, it allows us to use q that is one bit shorter than that used in Frodo. Briefly speaking, by carefully choosing parameters LWE-based KE protocols from OKCN and AKCN can be more efficient, more secure, of double length of consensus bits (with low extra cost); or have significantly lower negligible failure probability on the same parameters of Frodo (for example, $2^{-105.9}$ vs. $2^{-38.9}$ for the recommended implementation) with very slight bandwidth expansion.
- For LWE-based key exchange from the instantiation of “OKCN simple” proposed in Algorithm 3 (page 15), which is referred to as OKCN-LWE for presentation simplicity, it is simpler than Frodo. Specifically, for OKCN-LWE, Con takes two bitwise operations, and Rec takes one subtraction (i.e., $\sigma_2 - v$) and two bitwise operations. More importantly, when implemented with the parameters as shown in Table 3, the size of matrix \mathbf{A} for OKCN-LWE is much shorter than that for Frodo as shown in Table 4. *We remark that a smaller matrix \mathbf{A} not only much saves the time cost in generating and sampling the matrix \mathbf{A} , but also and more importantly, much improves the efficiency of matrix operations.* The similar hold for LWE-based key exchange from “AKCN power 2” proposed in Algorithm 5, referred to as AKCN-LWE.

In this section, we focus on the performance of OKCN-LWE without using the technique of cutting off some least significant bits, i.e., $t = 0$ in Figure 5. The performance of OKCN-LWE with $t > 0$ is analyzed in Section 5.4. We focus on the implementations of OKCN-LWE on two sets of parameters, with different optimization goals. The first set of parameters, w.r.t. discrete

	q	n	l	m	g		d		dist.	error rates		bw. (kB)	$ A $ (kB)	$ K $
					OKCN	Frodo	OKCN	Frodo		OKCN	Frodo			
Challenge	2^{10}	334	8	2^1	2^9	2	255	127	D_1	$2^{-47.9}$	$2^{-14.9}$	6.75	139.45	64
Classical	2^{11}	554	8	2^2	2^9	2	255	127	D_2	$2^{-39.4}$	$2^{-11.5}$	12.26	422.01	128
Recommended	2^{14}	718	8	2^4	2^{10}	2	511	255	D_3	$2^{-37.9}$	$2^{-10.2}$	20.18	902.17	256
Paranoid	2^{14}	818	8	2^4	2^{10}	2	511	255	D_4	$2^{-32.6}$	$2^{-8.6}$	22.98	1170.97	256
Paranoid-512	2^{12}	700	16	2^2	2^{10}	2	511	255	\bar{D}_4	$2^{-33.6}$	$2^{-8.3}$	33.92	735.00	512

Table 3: Parameters proposed for OKCN-LWE. “distr.” refers to the discrete distributions proposed in Table 1 and Table 2. “bw.” means bandwidth. $|A|$ refers to the size of the matrix. $|K| = l^2 \log m$ denotes the length of consensus bits.

	q	n	l	m	g		d		dist.	error rates		bw. (kB)		$ A $ (kB)	$ K $
					OKCN	Frodo	OKCN	Frodo		OKCN	Frodo	OKCN	Frodo		
Challenge	2^{11}	352	8	2^1	2^2	2	383	255	D_1	$2^{-80.1}$	$2^{-41.8}$	7.76	7.75	170.37	64
Classical	2^{12}	592	8	2^2	2^2	2	383	255	\bar{D}_2	$2^{-70.3}$	$2^{-36.2}$	14.22	14.22	525.70	128
Recommended	2^{15}	752	8	2^4	2^3	2	895	511	D_3	$2^{-105.9}$	$2^{-38.9}$	22.58	22.57	1060.32	256
Paranoid	2^{15}	864	8	2^4	2^3	2	895	511	\bar{D}_4	$2^{-91.9}$	$2^{-33.8}$	25.94	25.93	1399.68	256

Table 4: Parameters of Frodo, and comparison with OKCN-LWE. Here, “distr.” refers to the discrete distributions specified in Table 2. Note that, on the parameters of Frodo, OKCN-LWE achieves significantly lower error rates, which are negligible and are thus sufficient for achieving CPA-secure public-key encryption schemes.

distributions specified in Table 1 and Table 2, is presented in Table 3, which is carefully chosen for better efficiency and stronger security while preserving comparable error rates. Note that “Paranoid-521”, w.r.t. the discrete distribution \bar{D}_4 , aims at reaching 512 consensus bits, with cost significantly lesser than that of double running a 256-bit protocol while still without essentially sacrificing security or failure probabilities. The second set of parameters, given in Table 4 w.r.t. the discrete distributions specified in Table 2, is just that proposed for Frodo in [BCD⁺16], on which OKCN-LWE achieves significantly lower failure probabilities (with very slight bandwidth expansion). Note that, for presentation simplicity, we take $l_A = l_B = l$ for the sets of parameters under consideration. Also, for space limitation, we use OKCN to denote OKCN-LWE in these tables. Each set of parameters consists of four categories of increasing security levels: challenge, classic, recommended, and paranoid. The evaluated security levels are summarized in Table 5 and Table 6 respectively.

The concrete error rates in Table 3 and 4, and the security levels in Table 5 and Table 6, for OKCN-LWE are calculated based on the codes from the open source project <https://github.com/lwe-frodo/parameter-selection>

provided by the Frodo team [BCD⁺16]. For security evaluation, similar to [ADPS16, BCD⁺16], we only consider the resistance to two kinds of BKZ attacks, specifically primal attack and dual attack [CN11] [SE94], with respect to the core-SVP hardness. The reader is referred to [ADPS16, BCD⁺16] for more details. The concrete security levels are calculated by running the same codes of Frodo. The error rates for OKCN-LWE in Table 3 and 4, are derived by computing $\Pr \left[|\Sigma_1[i, j] - \Sigma_2[i, j]|_q > d \right]$, for any $1 \leq i \leq l_A$ and $1 \leq j \leq l_B$, and then applying the union bound. The concrete failure probabilities are gotten by running the codes slightly adjusted, actually simplified, from the open source codes of Frodo (see more details in Appendix A).

Scheme	Attack	Rounded Gaussian					Post-reduction		
		m'	b	C	Q	P	C	Q	P
Challenge	Primal	327	275	–	–	–	–	–	–
	Dual	310	272	–	–	–	–	–	–
Classical	Primal	477	444	138	126	100	132	120	95
	Dual	502	439	137	125	99	131	119	94
Recommended	Primal	664	500	155	141	112	146	133	105
	Dual	661	496	154	140	111	145	132	104
Paranoid	Primal	765	586	180	164	130	179	163	130
	Dual	743	582	179	163	129	178	162	129
Paranoid-512	Primal	643	587	180	164	131	180	164	130
	Dual	681	581	179	163	129	178	162	129

Table 5: Security estimation of the parameters described in Table 3. “C, Q, P” stand for “Classical, Quantum, Plausible” respectively. Numbers under these columns are the binary logarithm of running time of the corresponding attacks. Numbers under “ m', b ” are the best parameters for the attacks. “Rounded Gaussian” refers to the ideal case that noises and errors follow the rounded Gaussian distribution. “Post-reduction” refers to the case of using discrete distributions as specified in Table 1.

Discussions on the comparisons between OKCN-LWE and Frodo. Let $q = 2^{\bar{q}}$, $g = 2^{\bar{g}}$ and $m = 2^{\bar{m}}$. As we see in Table 4, on the same set of parameters, the bandwidth of OKCN-LWE is only very slightly larger than that of Frodo, but the error rate of OKCN-LWE is significantly lower than that of Frodo. The slight bandwidth expansion of OKCN-LWE is due to that the length of the signal value v in OKCN-LWE is \bar{g} but 1 in Frodo. However, as shown in Table 3, bandwidth loss caused by a longer v with OKCN-LWE in Table 4 is

overwhelmed by the gain of using a one-bit shorter q in Table 3. This is allowed by the tighter constraint $2md < q$ enjoyed by OKCN-LWE, compared to $4md < q$ with Frodo. Beyond saving bandwidth, employing a one-bit shorter q also much improves the computational efficiency (as the matrix \mathbf{A} becomes shorter, and consequently the cost of generating \mathbf{A} and the related matrix operations are more efficient), and can render stronger security levels simultaneously. Moreover, as we see from Table 3, 512 consensus bits can be achieved with OKCN-LWE, with the following advantages simultaneously (compared to Paranoid implementation of Frodo): stronger security, a much shorter matrix \mathbf{A} (which dominates computational efficiency), and insignificant 30.81% bandwidth expansion, and essentially the same error rate. This is much more economic than running a 256-bit protocol twice. Note that, on the parameters specified in Table 3, the error rates of Frodo are too large to be considered practical.

Scheme	Attack	Rounded Gaussian					Post-reduction		
		m'	b	C	Q	P	C	Q	P
Challenge	Primal	338	266	–	–	–	–	–	–
	Dual	331	263	–	–	–	–	–	–
Classical	Primal	549	442	138	126	100	132	120	95
	Dual	544	438	136	124	99	130	119	94
Recommended	Primal	716	489	151	138	110	145	132	104
	Dual	737	485	150	137	109	144	130	103
Paranoid	Primal	793	581	179	163	129	178	162	129
	Dual	833	576	177	161	128	177	161	128

Table 6: Security estimation of the parameters proposed for Frodo in [BCD⁺16], as specified in Table 4.

5.4 Integration into liboqs, and Benchmark

[SM16] introduces the Open Quantum Safe Project, which includes liboqs, a C library of quantum-resistant algorithms, and provides integration of liboqs into open-source applications and protocols including OpenSSL.

We fork the open source project liboqs on Github and add our OKCN-LWE scheme <http://github.com/OKCN>. Most of the codes are modified from Frodo-Recommended provided in liboqs.

Considering potential combinational attacks, we set the lower bound of σ to 1, as suggested in [Pop16]. The lower bound we set on σ prevent us

dist.	bits	var.	probability of							order	divergence
			0	± 1	± 2	± 3	± 4	± 5			
D_5	16	1.30	22218	15490	5242	858	67	2	500.0	1.0000337	

Table 7: Discrete distributions used in this work, and their Rényi divergences.

	q	n	l	m	g	t	d	dist.	err.	bw. (kB)	$ A $ (kB)	$ K $
Recommended	2^{14}	712	8	2^4	2^8	2	509	D_5	$2^{-39.0}$	18.58	887.15	256
Recommended-Enc	2^{14}	712	8	2^4	2^8	1	509	D_5	$2^{-52.3}$	19.29	887.15	256

Table 8: Parameters proposed for OKCN-LWE-Recommended. “distr.” refers to the discrete distributions proposed in Table 7. “bw.” means bandwidth. $|A|$ refers to the size of the matrix. $|K| = l^2 \log m$ denotes the length of consensus bits. “Recommended-Enc” is a parameter set with error rates that is low enough for public key encryption.

from choosing very small q . Hence, we cut off some least significant bits of \mathbf{Y}_2 to save bandwidth in practice.

Scheme	Attack	Rounded Gaussian					Post-reduction		
		m'	b	C	Q	P	C	Q	P
Recommend	Primal	638	480	149	136	108	148	135	107
	Dual	640	476	148	135	107	147	134	106

Table 9: Security estimation of the parameters described in Table 8. “C, Q, P” stand for “Classical, Quantum, Plausible” respectively. Numbers under these columns are the binary logarithm of running time of the corresponding attacks. Numbers under “ m', b ” are the best parameters for the attacks. “Rounded Gaussian” refers to the ideal case that noises and errors follow the rounded Gaussian distribution. “Post-reduction” refers to the case of using discrete distributions as specified in Table 7.

	iter.	total(s)	time(us)	stdev	cycle	stdev	bw. (B)
RLWE BCNS15							
Alice 0	959	1.001	1043.445	6.167	2394224	14052	4096
Bob	600	1.000	1666.755	8.125	3824541	18610	4244
Alice 1	4654	1.000	214.911	2.819	493007	6364	-
RLWE NewHope							
Alice 0	11712	1.000	85.390	1.922	195840	4308	1824
Bob	7828	1.000	127.762	1.938	293068	4297	2048
Alice 1	38061	1.000	26.274	1.071	60191	2254	-
LWE Frodo recommended							
Alice 0	728	1.000	1373.764	6.787	3152258	15542	11280
Bob	520	1.001	1924.763	9.663	4416649	22201	11288
Alice 1	6892	1.000	145.101	2.409	332845	5473	-
LWE OKCN recommended							
Alice 0	796	1.000	1256.353	8.780	2882846	20101	9968
Bob	577	1.001	1735.312	8.339	3981877	19153	8608
Alice 1	8114	1.000	123.252	2.244	282715	5095	-

Table 10: Benchmark of liboqs integrated with OKCN-LWE-Recommended. “iter.” refers to number of iterations. “time(us)” refers to mean time that spent on each iteration. ”cycle” refers to mean number of cpu cycles. “stdev” refers to population standard deviation of time or cpu cycles. “bw. (B)” refers to bandwidth, counted in bytes.

We run benchmark of liboqs on Ubuntu 14.04.5, gcc 4.9.2, Intel Core i7-4712MQ 2.30GHz, with hyperthreading and TurboBoost disabled, and the CPU frequency fixed to 2.30GHz (by following the instructions on <http://bench.cr.yp.to/supercop.html>). The benchmark result (Table 10) shows that OKCN-LWE-Recommended is slightly faster than Frodo, and uses smaller bandwidth.

6 RLWE-Based Key Exchange from KC and AKC

Denote by $(\lambda, n, q, \alpha, KC)$ the system parameters, where λ is the security parameter, $q \geq 2$ is a positive prime number, α parameterizes the discrete Gaussian distribution $D_{\mathbb{Z}^n, \alpha}$, n denotes the degree of polynomials in \mathcal{R}_q , and Gen a PRG generating $\mathbf{a} \in \mathcal{R}_q$ from a small seed. The length of random seed, i.e., κ , is typically set to be 256.

Let $KC = (\text{params}, \text{Con}, \text{Rec})$ be a correct and secure KC scheme, where $\text{params} = (q, g, m, d)$. The KC-based key exchange protocol from RLWE is

depicted in Figure 4, where the actual session-key is derived from \mathbf{K}_1 and \mathbf{K}_2 via some key derivation function KDF that may be modelled as a random oracle. There, for presentation simplicity, the Con and Rec functions are applied to polynomials, meaning they are applied to each of the coefficients respectively. Note that $\sigma_1 = \mathbf{y}_2 \cdot \mathbf{x}_1 = \mathbf{a} \cdot \mathbf{x}_2 \cdot \mathbf{x}_1 + \mathbf{e}_2 \cdot \mathbf{x}_1$, $\sigma_2 = \mathbf{y}_1 \cdot \mathbf{x}_2 + \mathbf{e}_\sigma = \mathbf{a} \cdot \mathbf{x}_1 \cdot \mathbf{x}_2 + \mathbf{e}_1 \cdot \mathbf{x}_2 + \mathbf{e}_\sigma$. As we choose $\mathbf{x}_1, \mathbf{x}_2, \mathbf{e}_1, \mathbf{e}_2$ according to a small noise distribution $D_{\mathbb{Z}^n, \alpha}$, the main part of σ_1 and that of σ_2 are the same $\mathbf{a} \cdot \mathbf{x}_1 \cdot \mathbf{x}_2$. Hence, the corresponding coordinates of σ_1 and σ_2 are close in the sense of $|\cdot|_q$, from which some key consensus can be reached. The error rate, i.e., failure probability, depends upon the concrete value of α and the distance parameter d . As discussed in Section 5, a KC-based key exchange protocol can be trivially extended to work on any correct and secure AKC scheme. As the bandwidth of RLWE-based KE protocol has already been low, we do not apply the technique of cutting off some least significant bits of each element of the polynomial \mathbf{y}_2 .

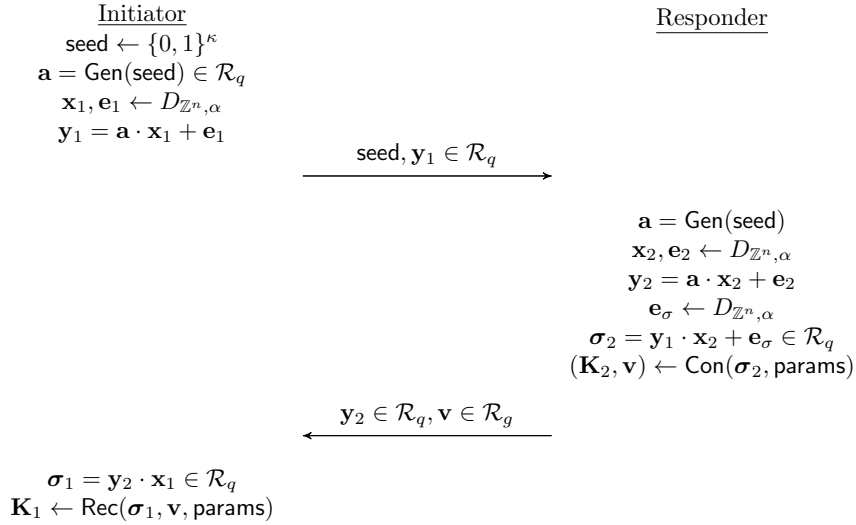


Figure 4: RLWE-based key exchange from KC and AKC, where $\mathbf{K}_1, \mathbf{K}_2 \in \mathcal{R}_q$. The protocol instantiated with OKCN specified in Algorithm 1 (resp., AKCN in Algorithm 4) is referred to as OKCN-RLWE (resp., AKCN-RLWE).

On security analysis. The security definition and proof of the RLWE-based key exchange protocol can be straightforwardly adapted from those, presented in Section 5.1, for the LWE-based KE protocol. Briefly speak-

ing, from Game G_0 to Game G_1 , we replace \mathbf{y}_1 with a uniformly random polynomial in \mathcal{R}_q . From Game G_1 to Game G_2 , we replace \mathbf{y}_2 and $\boldsymbol{\sigma}_2$ with uniformly random polynomials. Then, from the *security* of the underlying KC or AKC, we get that for any i , $1 \leq i \leq n$, $\mathbf{K}_2[i]$ and $\mathbf{v}[i]$ are independent, and so the protocol is secure.

On implementations of RLWE-based KE. We remark that most of the speeding-up techniques, discussed in Section 5, for LWE-based KE can also be applied here. The protocol described in Figure 4 works on any hard instantiation of the RLWE problem. But if n is power of 2, and prime q satisfies $q \bmod 2n = 1$, then number-theoretic transform (NTT) can be used to speed up polynomial multiplication. The performance can be further improved by using the Montgomery arithmetic and AVX2 instruction set [ADPS16], and by carefully optimizing performance-critical routines (in particular, NTT) in ARM assembly [AJS16].

For all the implementations considered in this work for RLWE-based KE, we use the same parameters and noise distributions proposed for NewHope in [ADPS16], as described in Table 11. They achieve about 281-bit (resp., 255-, 199-) security against classic (resp., quantum, plausible) attacks. The reader is referred to [ADPS16] for details. In particular, the underlying noise distribution is the centered binomial distribution Ψ_{16} (rather than rounded Gaussian distribution with the standard deviation $\sqrt{8}$), which can be rather trivially sampled in hardware and software with much better protection against timing attacks.

6.1 AKCN-RLWE with Negligible Error Rate

When implemented with the same parameters proposed in [ADPS16] for NewHope, as shown in Table 11 OKCN-RLWE and AKCN-RLWE reach 1024 consensus bits, with a failure probability around 2^{-40} that, we suggest, suffices for most applications of key exchange. In order for reaching a negligible error rate, particularly for achieving a CPA-secure PKE scheme, we also develop a variant of AKCN, referred to as AKCN-4:1, by borrowing some ideas from NewHope.

6.1.1 Overview of NewHope

By extending the technique of [PG13], in NewHope the coefficients of $\boldsymbol{\sigma}_1$ (i.e., the polynomial of degree n) are divided into $n/4$ groups, where each group contains four coordinates. On the input of four coordinates, only one

bit (rather than four bits) consensus is reached, which reduces the error rate to about 2^{-61} which is viewed to be negligible in practice.

Specifically, suppose Alice and Bob have σ_1 and σ_2 in \mathbb{Z}_q^4 respectively, and they are close to each other. One can regard the two vectors as elements in $\mathbb{R}^4/\mathbb{Z}^4$, by treating them as $\frac{1}{q}\sigma_1$ and $\frac{1}{q}\sigma_2$. Consider the matrix $\mathbf{B} = (\mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \mathbf{g}) \in \mathbb{R}^{4 \times 4}$, where \mathbf{u}_i , $0 \leq i \leq 2$, is the canonical unit vector whose i -th coordinate is 1, and $\mathbf{g} = (1/2, 1/2, 1/2, 1/2)^T$. Denote by \tilde{D}_4 the lattice generated by \mathbf{B} . Note that $\mathbb{Z}^4 \subset \tilde{D}_4 \subset \mathbb{R}^4$. Denote by \mathcal{V} the close Voronoi cell of the origin in \tilde{D}_4 . In fact, \mathcal{V} is the intersection of the unit ball in norm 1 and the unit ball in infinity norm (the reader is referred to NewHope [ADPS16, Appendix C] for details). The following procedure $\text{CVP}_{\tilde{D}_4}(\mathbf{x})$ returns the vector \mathbf{v} such that $\mathbf{B}\mathbf{v}$ is closest to \mathbf{x} , i.e., $\mathbf{x} \in \mathbf{B}\mathbf{v} + \mathcal{V}$, where the distance is measured in the Euclidean norm.

Algorithm 13 $\text{CVP}_{\tilde{D}_4}$ in NewHope [ADPS16]

```

1: procedure  $\text{CVP}_{\tilde{D}_4}(\mathbf{x} \in \mathbb{R}^4)$ 
2:    $\mathbf{v}_0 = \lfloor \mathbf{x} \rfloor$ 
3:    $\mathbf{v}_1 = \lfloor \mathbf{x} - \mathbf{g} \rfloor$ 
4:    $k = 0$  if  $\|\mathbf{x} - \mathbf{v}_0\|_1 < 1$  and 1 otherwise
5:    $(v_0, v_1, v_2, v_3)^T = \mathbf{v}_k$ 
6:   return  $\mathbf{v} = (v_0, v_1, v_2, k)^T + v_3 \cdot (-1, -1, -1, 2)^T$ 
7: end procedure
```

If σ_1 is in the Voronoi cell of \mathbf{g} , then the consensus bit is set to be 1, and 0 otherwise. Hence, Alice finds the closest lattice vector of σ_1 by running the $\text{CVP}_{\tilde{D}_4}$ procedure described in Algorithm 13, and calculates their difference which is set to be the hint signal \mathbf{v} . Upon receiving \mathbf{v} , Bob subtracts the difference from σ_2 . Since σ_1 and σ_2 are very close, the subtraction moves $\frac{1}{q}\sigma_2$ towards a lattice point in \tilde{D}_4 . Then Bob checks whether or not the point after the move is in the Voronoi cell of \mathbf{g} , and so the consensus is reached. Furthermore, to save bandwidth, NewHope chooses an integer r , and discretizes the Voronoi cell of \mathbf{g} to 2^{4r} blocks, so that only $4r$ bits are needed to transfer the hint information. To make the distribution of consensus bit uniform, NewHope adds a small noise to σ_1 , similar to the dbl trick used in [Pei14]. The Con and Rec procedures, distilled from NewHope, are presented in Algorithm 16 in Appendix C.

6.1.2 Construction and Analysis of AKCN-4:1

For any integer q and vector $\mathbf{x} = (x_0, x_1, x_2, x_3)^T \in \mathbb{Z}_q^4$, denote by $\|\mathbf{x}\|_{q,1}$ the sum $|x_0|_q + |x_1|_q + |x_2|_q + |x_3|_q$. For two vectors $\mathbf{a} = (a_0, a_1, a_2, a_3)^T, \mathbf{b} = (b_0, b_1, b_2, b_3)^T \in \mathbb{Z}^4$, let $\mathbf{a} \bmod \mathbf{b}$ denote the vector $(a_0 \bmod b_0, a_1 \bmod b_1, a_2 \bmod b_2, a_3 \bmod b_3)^T \in \mathbb{Z}^4$. The scheme of AKCN-4:1 is presented in Algorithm 14.

Compared with the consensus mechanism of NewHope presented in Appendix C, AKCN-4:1 can be simpler and computationally more efficient. In specific, the uniformly random bit b used in NewHope (corresponding the dbl trick in [Pei14]) is eliminated with AKCN-4:1, which saves 256 (resp., 1024) random bits in total when reaching 256 (resp., 1024) consensus bits. In addition, as k_1 , as well as $k_1(q+1)\mathbf{g}$, can be offline computed and used (e.g., for encryption, in parallel with the protocol run), AKCN-4:1 enjoys online/offline speeding-up and parallel computing.

Theorem 6.1. *If $\|\sigma_1 - \sigma_2\|_{q,1} < q \left(1 - \frac{1}{g}\right) - 2$, then the AKCN-4:1 scheme depicted in Algorithm 14 is correct.*

Proof. Suppose $\mathbf{v}' = \text{CVP}_{\tilde{D}_4}(g(\sigma_1 + k_1(q+1)\mathbf{g})/q)$. Then, $\mathbf{v} = \mathbf{v}' \bmod (g, g, g, 2g)$, and so there exists $\boldsymbol{\theta} = (\theta_0, \theta_1, \theta_2, \theta_3) \in \mathbb{Z}^4$ such that $\mathbf{v} = \mathbf{v}' + g(\theta_0, \theta_1, \theta_2, 2\theta_3)^T$. From the formula calculating \mathbf{v}' , we know there exists $\boldsymbol{\epsilon} \in \mathcal{V}$, such that $g(\sigma_1 + k_1(q+1)\mathbf{g})/q = \boldsymbol{\epsilon} + \mathbf{B}\mathbf{v}'$. Hence, $\mathbf{B}\mathbf{v}' = g(\sigma_1 + k_1(q+1)\mathbf{g})/q - \boldsymbol{\epsilon}$.

From the formula computing \mathbf{x} in Rec, we have $\mathbf{x} = \mathbf{B}\mathbf{v}/g - \sigma_2/q = \mathbf{B}\mathbf{v}'/g - \sigma_2/q + \mathbf{B}(\theta_0, \theta_1, \theta_2, 2\theta_3)^T = k_1\mathbf{g} + k_1\mathbf{g}/q - \boldsymbol{\epsilon}/g + (\sigma_1 - \sigma_2)/q + \mathbf{B}(\theta_0, \theta_1, \theta_2, 2\theta_3)^T$. Note that the last term $\mathbf{B}(\theta_0, \theta_1, \theta_2, 2\theta_3)^T \in \mathbb{Z}^4$, and in line 7 of Algorithm 14 we subtract $\lfloor \mathbf{x} \rfloor \in \mathbb{Z}^4$ from \mathbf{x} , so the difference between $\mathbf{x} - \lfloor \mathbf{x} \rfloor$ and $k_1\mathbf{g}$ in norm 1 is no more than $2/q + 1/g + \|\sigma_1 - \sigma_2\|_{q,1}/q < 1$. Hence, $k_2 = k_1$. \square

Algorithm 14 AKCN-4:1

```

1: procedure CON( $\sigma_1 \in \mathbb{Z}_q^4, k_1 \in \{0, 1\}, \text{params}$ )
2:    $\mathbf{v} = \text{CVP}_{\tilde{D}_4}(g(\sigma_1 + k_1(q+1)\mathbf{g})/q) \bmod (g, g, g, 2g)^T$ 
3:   return  $\mathbf{v}$ 
4: end procedure
5: procedure REC( $\sigma_2 \in \mathbb{Z}_q^4, \mathbf{v} \in \mathbb{Z}_g^3 \times \mathbb{Z}_{2g}, \text{params}$ )
6:    $\mathbf{x} = \mathbf{B}\mathbf{v}/g - \sigma_2/q$ 
7:   return  $k_2 = 0$  if  $\|\mathbf{x} - \lfloor \mathbf{x} \rfloor\|_1 < 1$ , 1 otherwise.
8: end procedure

```

Theorem 6.2. *AKCN-4:1 depicted in Algorithm 14 is secure. Specifically, if σ_1 is subject to uniform distribution over \mathbb{Z}_q^4 , then \mathbf{v} and k_1 are independent.*

Proof. Let $\mathbf{y} = (\sigma_1 + k_1(q+1)\mathbf{g}) \bmod q \in \mathbb{Z}_q^4$. First we prove that \mathbf{y} is independent of k_1 , when $\sigma_1 \leftarrow \mathbb{Z}_q^4$. Specifically, for arbitrary $\tilde{\mathbf{y}} \in \mathbb{Z}_q^4$ and arbitrary $\tilde{k}_1 \in \{0, 1\}$, we want to prove that $\Pr[\mathbf{y} = \tilde{\mathbf{y}} \mid k_1 = \tilde{k}_1] = \Pr[\sigma_1 = (\tilde{\mathbf{y}} - k_1(q+1)\mathbf{g}) \bmod q \mid k_1 = \tilde{k}_1] = 1/q^4$. Hence, \mathbf{y} and k_1 are independent.

For simplicity, denote by \mathbf{G} the vector $(g, g, g, 2g)$. Map $\phi : \mathbb{Z}^4 \rightarrow \mathbb{Z}_g^3 \times \mathbb{Z}_{2g}$ is defined by $\phi(\mathbf{w}) = \text{CVP}_{\tilde{D}_4}(g\mathbf{w}/q) \bmod \mathbf{G}$. We shall prove that, for any $\boldsymbol{\theta} \in \mathbb{Z}^4$, $\phi(\mathbf{w} + q\boldsymbol{\theta}) = \phi(\mathbf{w})$. By definition of ϕ , $\phi(\mathbf{w} + q\boldsymbol{\theta}) = \text{CVP}_{\tilde{D}_4}(g\mathbf{w}/q + g\boldsymbol{\theta}) \bmod \mathbf{G}$. Taking $\mathbf{x} = g\mathbf{w}/q + g\boldsymbol{\theta}$ into Algorithm 13, we have $\text{CVP}_{\tilde{D}_4}(g\mathbf{w}/q + g\boldsymbol{\theta}) = \text{CVP}_{\tilde{D}_4}(g\mathbf{w}/q) + \mathbf{B}^{-1}(g\boldsymbol{\theta})$. It is easy to check that the last term $\mathbf{B}^{-1}(g\boldsymbol{\theta})$ always satisfies $\mathbf{B}^{-1}(g\boldsymbol{\theta}) \bmod \mathbf{G} = 0$.

From the above property of ϕ , we have $\phi(\mathbf{y}) = \phi((\sigma_1 + k_1(q+1)\mathbf{g}) \bmod q) = \phi(\sigma_1 + k_1(q+1)\mathbf{g}) = \mathbf{v}$. As k_1 is independent of \mathbf{y} , and \mathbf{v} only depends on \mathbf{y} , k_1 and \mathbf{v} are independent. \square

6.2 Instantiations, and Comparison with NewHope

	q	n	m	g	d	distr.	$ \mathbf{K} $	bw. (B)	err.
OKCN-RLWE	12289	1024	2^1	2^4	2879	Ψ_{16}	1024	4128	2^{-38}
OKCN-RLWE	12289	1024	2^1	2^6	3023	Ψ_{16}	1024	4384	2^{-42}
AKCN-RLWE	12289	1024	2^1	2^4	2687	Ψ_{16}	1024	4128	2^{-32}
AKCN-RLWE	12289	1024	2^1	2^6	2975	Ψ_{16}	1024	4384	2^{-41}
NewHope	12289	1024	-	2^2	-	Ψ_{16}	256	3872	2^{-61}
AKCN-4:1-RLWE	12289	1024	-	2^2	-	Ψ_{16}	256	3872	2^{-61}

Table 11: Comparisons with NewHope. distr. is the noise distributions, and Ψ_{16} is the sum of 16 independent centered binomial variables [ADPS16]. $|\mathbf{K}|$ refers to the total binary length of consensus bits. bw. (B) refers to the bandwidth in bytes. err. refers to failure probability.

The comparisons between NewHope and RLWE-based KE protocols from OKCN as specified in Algorithm 1 (referred to as OKCN-RLWE), from AKCN as specified in Algorithm 4 (referred to as AKCN-RLWE) and from AKCN-4:1 as specified in Algorithm 14 (referred to as AKCN-4:1-RLWE), are presented in Table 11. The comparisons are made on the same parameters (q, n) and the same noise distribution as proposed in [ADPS16] for Newhope, and so all the instantiations of our schemes have the same security level of Newhope.

In Table 11, the error rates of OKCN-RLWE and AKCN-RLWE are obtained by using the adjusted codes of Frodo as discussed in Appendix A, with the noise distribution being changed to the centered binomial distribution Ψ_{16} for NewHope. As the correctness condition for AKCN-4:1 (Theorem 6.1) and that for NewHope (Lemma C.3 in [ADPS16]) are the same, they have the same error rate.

For OKCN/AKCN-RLWE, the number of consensus bits is 4 times of that of NewHope, though the bandwidth is expanded about 6.6% or 13.2% and error rate is increased to around 2^{-40} that, we suggest, is still reasonable for most applications. For AKCN-4:1-RLWE, it has the same error rate as NewHope, but it enjoys the advantages of session-key predetermination and thus great online/offline and parallel computability. We remark that a direct instantiation of RLWE-based KE protocol from the AKCN-4:1 scheme described in Algorithm 14, its bandwidth is 256-bit longer than that of NewHope, since the last coordinate of \mathbf{v} needs 3 bits to represent with AKCN-4:1. However, such 256-bit expansion in bandwidth can be avoided by a more compact encode method, because NewHope uses 14 bits to encode an element in \mathbb{Z}_q ; as $2q^3 < 2^{14 \times 3}$, one can encode one more bit every three elements in \mathbb{Z}_q^3 . In addition, compared to NewHope, OKCN/AKCN-RLWE (resp., AKCN-4:1-RLWE) avoids sampling 1024 (resp., 256) random bits for Con.

7 Applications to PKE, OT, Key Transport, and TLS 1.3

It is well known that the composition of a secure KE protocol (with negligible error rate) and a CPA-secure symmetric-key encryption scheme yields a CPA-secure PKE scheme. And any CPA-secure PKE can be transformed into a CCA-secure one via the FO-transformation [FO13, FO99, Pei14, 1] in the random oracle model. If we view 2^{-60} to be negligible, then OKCN-LWE and AKCN-LWE (on the same parameters of Frodo) and AKCN-4:1-RLWE (on the same parameters of NewHope) can be used to build CPA-secure PKE schemes. Moreover, AKCN-LWE and AKCN-4:1-RLWE can be used alone for CPA-secure PKE scheme to encrypt 256-bit messages. To the best of our knowledge, they lead to the state-of-the-art practical schemes of CPA-secure PKE, as well as CCA-secure PKE in the random oracle model, from LWE and RLWE.

Oblivious transfer (OT) is another fundamental primitive of cryptography. We say a CPA-secure PKE scheme is PK-samplable, if the distribution

of the public key is indistinguishable from some distribution, e.g., uniform distribution, that can be sampled efficiently without the knowledge of the secret key. Clearly, all the CPA-secure PKE schemes from OKCN/AKCN-LWE and AKCN-4:1-RLWE are PK-samplable under the LWE or RLWE assumption. It is shown in [GKM⁺00] that any PK-samplable CPA-secure PKE scheme can be used, as a black box, to construct an OT protocol with honest yet curious parties, which can be further transformed into an OT protocol against any PPT malicious users by employing zero-knowledge proof.

One particularly important application of public-key cryptography is key transport (i.e., public-key encryption of a random symmetric key), which is in particular demanded by the Tor project [Nic] and NIST [NIS]. We note that our AKC-based KE protocols can just be used for key transport.

Any secure KE protocol can be transformed, in a black-box way, into an authenticated key exchange (AKE) protocol by additionally using a secure signature scheme via the SIGMA paradigm [Kra03]. SIGMA is just the basis of authentication mechanism for the secure transport protocol TLS in the client/server setting. Recently, the next generation of TLS, specifically TLS1.3, is now under development [Res]; And developing post-quantum secure TLS protocol is now receiving more and more efforts or attention both from research community and from standardization agencies. Compared to the currently deployed TLS1.2 standard, one salient change (among others) made in TLS1.3 is that the server now plays the role of the responder. The heavy workload in the server, particularly at peak time, is one of the major sources that causes slower server responsiveness or causes the server an easy target of more and more vicious DDoS attacks. We suggest that the predicament faced by the server can be mitigated with AKC-based KE protocols like AKCN-LWE and AKCN-4:1-RLWE, by taking advantage of the session-key predetermination and online/offline parallel computability enjoyed by them.

Acknowledgement: We are grateful to Leixiao Cheng, Boru Gong and Qin Luo for helpful discussions.

References

- [ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. *Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems*. Springer Berlin Heidelberg, 2009.

- [ADPS16] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum Key Exchange — A New Hope. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 327–343, Austin, TX, August 2016. USENIX Association.
- [AG11] Sanjeev Arora and Rong Ge. New Algorithms for Learning in Presence of Errors. In *International Colloquium on Automata, Languages, and Programming*, pages 403–415. Springer, 2011.
- [AJS16] Erdem Alkim, Philipp Jakubeit, and Peter Schwabe. A new hope on arm cortex-m. Cryptology ePrint Archive, Report 2016/758, 2016. <http://eprint.iacr.org/2016/758>.
- [BGMRR16] A. Bogdanov, S. Guo, D. Masny, S. Richelson, and A. Rosen. On the hardness of learning with rounding over small modulus. TCC 2016: 209-224.
- [BCD⁺16] Joppe Bos, Craig Costello, Lo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. Frodo: Take off the ring! Practical, Quantum-Secure Key Exchange from LWE. In *23rd ACM Conference on Computer and Communications Security (ACM CCS)*, 2016.
- [BCNS15] Joppe W. Bos, Craig Costello, Michael Naehrig, and Douglas Stebila. Post-Quantum Key Exchange for the TLS Protocol from the Ring Learning with Errors Problem. In *Proceedings of the 2015 IEEE Symposium on Security and Privacy*, SP '15, pages 553–570, Washington, DC, USA, 2015. IEEE Computer Society.
- [BLL⁺15] Shi Bai, Adeline Langlois, Tancrede Lepoint, Damien Stehlé, and Ron Steinfeld. Improved security proofs in lattice-based cryptography: using the Rényi divergence rather than the statistical distance. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 3–24. Springer, 2015.
- [CESG] CESG. Quantum key distribution: A CESG white paper, 2016. <https://www.cesg.gov.uk/white-papers/quantum-key-distribution>.
- [CN11] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better Lattice Security Estimates. In *Advances in Cryptology - ASIACRYPT*

2011 - *International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, pages 1–20, 2011.

- [CHKLS16] J.H. Cheon, K.H. Han, J. Kim, C.Lee, and Y. Song. A Practical Post-Quantum Public-Key Cryptosystem Based on spLWE. Cryptology ePrint Archive, Report 2016/1055, 2016. <http://eprint.iacr.org/2016/1055>.
- [CKLS16] J.H. Cheon, D. Kim, Joohee Lee, and Y. Song. Lizard: Cut Off the Tail! Practical Post-Quantum Public-Key Encryption from LWE and LWR. Cryptology ePrint Archive, Report 2016/1126, 2016. <http://eprint.iacr.org/2016/1126>.
- [CW90] Don Coppersmith and Shmuel Winograd. Matrix Multiplication via Arithmetic Progressions. *Journal of Symbolic Computation*, 9(3):251–280, 1990.
- [DD12] Léo Ducas and Alain Durmus. *Ring-LWE in Polynomial Rings*, pages 34–51. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.
- [FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. *How to Enhance the Security of Public-Key Encryption at Minimum Cost*, pages 53–68. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.
- [FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, 2013.
- [GKM⁺00] Y. Gertner, S. Kannan, T. Malkin, O. Reingold, and M. Viswanathan. The Relationship Between Public Key Encryption and Oblivious Transfer. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, FOCS’00, pages 325–, Washington, DC, USA, 2000. IEEE Computer Society.
- [GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for Hard Lattices and New Cryptographic Constructions. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC ’08, pages 197–206, New York, NY, USA, 2008. ACM.

- [GS16] Shay Gueron and Fabian Schlieker. Speeding up R-LWE Post-Quantum Key Exchange. Cryptology ePrint Archive, Report 2016/467, 2016. <http://eprint.iacr.org/2016/467>.
- [JD12] Xiaodong Lin Jintai Ding, Xiang Xie. A Simple Provably Secure Key Exchange Scheme Based on the Learning with Errors Problem. Cryptology ePrint Archive, Report 2012/688, 2012. <http://eprint.iacr.org/2012/688>.
- [KF15] Paul Kirchner and Pierre-Alain Fouque. An improved BKW algorithm for LWE with applications to cryptography and lattices. In *Annual Cryptology Conference*, pages 43–62. Springer, 2015.
- [Kra03] Hugo Krawczyk. *SIGMA: The ‘SIGn-and-MAC’ Approach to Authenticated Diffie-Hellman and Its Use in the IKE Protocols*, pages 400–425. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.
- [LP10] Richard Lindner and Chris Peikert. Better Key Sizes (and Attacks) for LWE-Based Encryption. 2010.
- [LPR13a] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On Ideal Lattices and Learning with Errors over Rings. *J. ACM*, 60(6):43:1–43:35, November 2013.
- [LPR13b] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. *A Toolkit for Ring-LWE Cryptography*, pages 35–54. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
- [Mat] Braithwaite Matt. Experimenting with post-quantum cryptography. Posting on the Google Security Blog, 2016. <https://security.googleblog.com/2016/07/experimenting-with-post-quantum.html>.
- [MP13] Daniele Micciancio and Chris Peikert. Hardness of SIS and LWE with small parameters. In *Advances in Cryptology–CRYPTO 2013*, pages 21–39. Springer, 2013.
- [Nic] Mathewson Nick. Cryptographic directions in Tor. Slides of a talk at Real-World Crypto 2016, 2016. <https://people.torproject.org/~nickm/slides/nickm-rwc-presentation.pdf>.

- [NIS] NIST. Post-Quantum Crypto Project. 2016. <http://csrc.nist.gov/groups/ST/post-quantum-crypto/index.html>.
- [NSA] NSA. NSA suite B cryptography. https://www.nsa.gov/ia/programs/suiteb_cryptography.
- [Pei14] Chris Peikert. Lattice Cryptography for the Internet. In *International Workshop on Post-Quantum Cryptography*, pages 197–219. Springer, 2014.
- [Pei16] Chris Peikert. A decade of lattice cryptography. *Foundations and Trends in Theoretical Computer Science*, 10(4):283–424, 2016.
- [PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. CRYPTO 2008: 554–571.
- [PG13] Thomas Pöppelmann and Tim Güneysu. Towards practical lattice-based public-key encryption on reconfigurable hardware. In *International Conference on Selected Areas in Cryptography*, pages 68–85. Springer, 2013.
- [Pop16] Alex van Poppel, Cryptographic decoding of the Leech lattice, Cryptology ePrint Archive, Report 2016/1050, 2016. <http://eprint.iacr.org/2016/1050>.
- [Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.
- [Res] Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3.
- [SE94] C. P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical Programming*, 66(1-3):181–199, 1994.
- [Str69] Volker Strassen. Gaussian Elimination is not Optimal. *Numerische Mathematik*, 13(4):354–356, 1969.
- [SM16] Douglas Stebila and Michele Mosca. Post-Quantum Key Exchange for the Internet and the Open Quantum Safe Project. Cryptology ePrint Archive, Report 2016/1017, 2016. <http://eprint.iacr.org/2016/1017>.

- [Pop16] Alex van Poppel, Cryptographic decoding of the Leech lattice, Cryptology ePrint Archive, Report 2016/1050, 2016. <http://eprint.iacr.org/2016/1050>.
- [1] E. E. Targhi and D. Unruh. Quantum security of the Fujisaki-Okamoto and OAEP transforms. Cryptology ePrint Archive, Report 2015/1210, 2015. <http://eprint.iacr.org/2015/1210>.

A On the Codes of Evaluating Error Rates of KE from OKCN and AKCN

For Frodo [BCD⁺16], if the distance between σ_1 (in the input of Con) and σ_2 (in the input of Rec), i.e., $d = |\sigma_1 - \sigma_2|_q$, is smaller than $q/4m$, then Frodo fails with probability 0. Otherwise, if the distance is larger than $3q/4m$, then Frodo fails with probability 1. For the distances between $q/4m$ and $3q/4m$, the failure probability of Frodo increase linearly from 0 to 1. In comparison, for OKCN-LWE, when the distance d is smaller than $q(1 - 1/g)/2m$ (i.e., $2md < q(1 - 1/g)$), it works perfectly. Otherwise, it fails.

To calculate the failure probability of OKCN-LWE, we modify (actually, simplifies) the function “pr_rec_failure()” in “failure_prob.py” provided by Frodo (<http://github.com/lwe-frodo/parameter-selection>) to only consider whether $2md < q(1 - 1/g)$ or not. The code adaptation is simple, which is only related to several lines of codes in “noise_failure_prob()” and is listed in Figure 5.

B Consensus Mechanism of Frodo

Let the modulo q be power of 2, which can be generalized to arbitrary modulo using the techniques in [Pei14]. Let integer B be a power of 2. $B < (\log q) - 1$, $\bar{B} = (\log q) - B$ (note that $m = 2^B$ in our notations). The underlying KC mechanism implicitly in Frodo is presented in Figure 15.

```

def noise_failure_prob(noise, q, n, w, reclen, g):
    m = 2**w
    d = max(filter(lambda x: m*2*x < q*(1. - 1./g), range(q)))
    print "d = ", d
    def pr_rec_failure(x):
        x = min(x, q - x)
        if x <= d:
            return 0
        else:
            return 1

    noise_sqr = pdf_product(noise, noise, q)
    v = nfoldconvolution(2 * n, noise_sqr, q)
    v = convolution(v, noise, q) # v = 2n * (noise^2) + noise
    exact_pr = {x: p * pr_rec_failure(x) for (x, p) in v.iteritems()}
    failure_pr = reclen * sum(exact_pr.itervalues())
    return failure_pr

```

Figure 5: Codes adjusted to error rate calculation of KE from OKC-N/AKCN.

Algorithm 15 Key consensus scheme in Frodo

```

1: procedure CON( $\sigma_1$ , params)  $\triangleright \sigma_1 \in [0, q]$ 
2:    $v = \left\lfloor 2^{-\bar{B}+1} \sigma_1 \right\rfloor \bmod 2$ 
3:    $k_1 = \left\lfloor 2^{-\bar{B}} \sigma_1 \right\rfloor \bmod 2^B$ 
4:   return ( $k_1, v$ )
5: end procedure
6: procedure REC( $\sigma_2, v$ , params)  $\triangleright \sigma_2 \in [0, q]$ 
7:   find  $x \in \mathbb{Z}_q$  closest to  $\sigma_2$  s.t.  $\left\lfloor 2^{-\bar{B}+1} x \right\rfloor \bmod 2 = v$ 
8:    $k_2 = \left\lfloor 2^{-\bar{B}} x \right\rfloor \bmod 2^B$ 
9:   return  $k_2$ 
10: end procedure

```

Claim B.1 ([BCD⁺16], Claim 3.2). *If $|\sigma_1 - \sigma_2|_q < 2^{\bar{B}-2}$, then $\text{Rec}(\sigma_2, v) = k_1$. i.e. the scheme in Algorithm 15 is correct.*

This claim is equivalence to require $4md < q$.

C Consensus Mechanism of NewHope

Note that, for the consensus mechanism of NewHope, the *rec* procedure is run both in *Con* and in *Rec*, and a random bit b is used in *Con* corresponding to the dbl trick in [Pei14].

Algorithm 16 NewHope Consensus Mechanism

```
1: procedure DECODE( $\mathbf{x} \in \mathbb{R}^4/\mathbb{Z}^4$ )       $\triangleright$  Return a bit  $k$  such that  $k\mathbf{g}$  is
   closest to  $\mathbf{x} + \mathbb{Z}^4$ 
2:    $\mathbf{v} = \mathbf{x} - \lfloor \mathbf{x} \rfloor$ 
3:   return  $k = 0$  if  $\|\mathbf{v}\|_1 \leq 1$ , and 1 otherwise
4: end procedure
5:
6: HelpRec( $\mathbf{x}, b$ ) = CVP $_{\tilde{D}_4} \left( \frac{2^r}{q}(\mathbf{x} + b\mathbf{g}) \right) \bmod 2^r \triangleright b$  corresponds to the dbl
   trick [Pei14]
7:  $rec(\mathbf{x} \in \mathbb{Z}_q^4, \mathbf{v} \in \mathbb{Z}_{2^r}^4) = \text{Decode} \left( \frac{1}{q}\mathbf{x} - \frac{1}{2^r}\mathbf{B}\mathbf{v} \right)$ 
8:
9: procedure CON( $\sigma_1 \in \mathbb{Z}_q^4$ , params)
10:   $b \leftarrow \{0, 1\}$ 
11:   $\mathbf{v} \leftarrow \text{HelpRec}(\sigma_1, b)$ 
12:   $k_1 \leftarrow rec(\sigma_1, \mathbf{v})$ 
13:  return  $(k_1, \mathbf{v})$ 
14: end procedure
15:
16: procedure REC( $\sigma_2 \in \mathbb{Z}_q^4, \mathbf{v} \in \mathbb{Z}_{2^r}^4$ , params)
17:   $k_2 \leftarrow rec(\sigma_2, \mathbf{v})$ 
18: end procedure
19:
```

D A Note on Lizard

We note that the CPA-secure PKE scheme, Lizard, proposed in [CKLS16] is actually instantiated from our AKCN scheme presented in Algorithm 4, where the two close values are derived from generating and exchanging spLWE and spLWR samples in an asymmetric way. Specifically, the public key is generated with spLWE samples, while ciphertext is generated with spLWR samples. However, the underlying AKC mechanism in the spLWE/spLWR based PKE scheme analyzed in [CKLS16] is actually an instantiation of our AKCN scheme for the special case of $m|g|q$, where g (resp., m) in AKCN corresponds to p (resp., t) in [CKLS16].